TAMPEREEN TEKNILLINEN YLIOPISTO

ARTO HÄRKÖNEN JA SAMI KOJO

TOSIAIKAISEN STRATEGIAPELIN KEHITTÄMINEN UNITY-PELINKEHITYSTYÖKALULLA

Diplomityö

Tarkastaja: prof. Tommi Mikkonen
Tarkastaja ja aihe hyväksytty Tieto- ja sähkötekniikan tiedekuntaneuvoston kokouksessa 5.2.2014.

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Tietotekniikan koulutusohjelma
HÄRKÖNEN, ARTO; KOJO, SAMI: Tosiaikaisen strategiapelin kehittäminen Unity-pelinkehitystyökalulla
Diplomityö, 65 sivua
Huhtikuu 2014
Pääaine: Ohjelmistotuotanto
Tarkastaja: professori Tommi Mikkonen
Avainsanat: Unity, tosiaikastrategia, tietokonepelit, reitinhakujärjestelmät, tietokonegrafiikka, A*, navigaatioverkko.

Fragment Production Oy perustettiin keväällä 2012, kun saksalainen julkaisija Rondomedia GmbH oli tilannut tietokonepelin Microsoft Windows -alustalle. Pelin teemaksi Rondomedia määritteli pelastusaiheisen simulaation ja tosiaikaisen strategian yhdistelmän. Peli keskittyy erityisesti palomiesten arkielämän kuvaamiseen. Projektin nimeksi valittiin Rescue Squad, joka oli alkuperäisen pelikonseptin nimi.

Projektin alkuperäinen aikataulu oli noin kymmenen kuukautta. Projektin kuluessa sen sisältöä päätettiin kasvattaa asiakkaan toimesta siten, että aikataulu venyi lopulta noin 15 kuukauteen. Lisäksi projektiin sisällytettiin julkaisun jälkeen kuuden kuukauden pituinen jälkituotantovaihe, jonka aikana kehitettiin eri markkinoille suunnattuja versioita.

Tässä diplomityössä käsitellään peliprojektin toteutusta käyttäen Unity Technologies -yrityksen Unity-pelinkehitystyökalua. Unity valittiin projektiin sen geneerisyyden vuoksi, ja koska projektin henkilöstöllä oli aiempaa kokemusta kyseisestä työkalusta. Työssä tarkastellaan projektin etenemistä välietappien määrittelyn ja toteutumisen kannalta, ja käydään läpi projektin aikana ilmenneitä suurimpia teknisiä ongelmia ja niiden ratkaisuja.

Unity osoittautui projektin onnistumisen kannalta hyväksi valinnaksi erityisesti sen geneerisyyden vuoksi. Unityn geneerisyys osoittautui myös ongelmaksi erityisesti reitinhakujärjestelmän osalta, ja koska se vaikeuttaa hyvän ohjelmistoarkkitehtuurityylin suunnittelua. Projektin ensimmäinen merkittävä etappi saavutettiin ajallaan, kun Saksan markkinoille suunnattu DVD-levylle painettu versio saatiin valmiiksi toukokuussa 2013. Projektin päätti joulukuussa 2013 julkaistu Steam-latauspalveluun kehitetty versio, jolla tavoitettiin myös Yhdysvaltojen suuret markkinat.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
HÄRKÖNEN, ARTO; KOJO, SAMI: Developing realtime strategy game with Unity
Master of Science Thesis, 65 pages
April 2014
Major: Software engineering
Examiner: Professor Tommi Mikkonen
Keywords: Unity, real-time strategy games, computer games, path-finding systems, computer graphics, A*, navigation mesh.

Fragment Production Ltd was founded in the Spring of 2012, when the German publisher Rondomedia GmbH ordered a computer game for the Microsoft Windows platform. Rondomedia defined the theme of the game to be centered around firefighting, and the genre to be real-time strategy. The game focuses especially on depicting the everyday life of rescue personnel. The project was named after the original Rescue Squad concept.

The original schedule for the project was about ten months. During the development, the project schedule was extended to 15 months due to the request of the client. After the initial release, a six month long post production phase was added to the project. During this time, additional localized versions were released to target different market areas.

This thesis focuses on the development of a game project using the Unity game development tool, produced by Unity Technologies. Unity was chosen due to its generic nature, and because the project personnel had previous experience with the tool. Additionally, the thesis describes defining the milestones and how they were then executed. Furthermore, the thesis explains some of the most challenging technical problems and their solutions.

Unity proved to be a good choice for the project due to its generic nature. However, the generality also presented problems, especially with the path-finding system, and because it makes defining the overall software architecture very difficult. The first major milestone was achieved according to schedule when the release version of the game was published on DVD in May 2013. The project ended in December 2013 when a digitally downloadable version of the game was released for the U.S. markets via Valve Corporation's Steam download service.

ALKUSANAT

Tämä diplomityö on tehty Fragment Production Oy:lle. Työ on tehty parityönä, jonka tekijät ovat Arto Härkönen ja Sami Kojo. Työssä toteutettiin tosiaikainen strategiapeli PC-tietokoneelle. Projektiin osallistui tämän työn kirjoittajien lisäksi useita muitakin henkilöitä.

Raportointityö jakautui pääasiallisesti projektin ja tietokonepelien taustoihin ja määrittelyihin sekä teknilliseen toteutusosaan. Näistä Arto Härkösen vastuulla oli projektin määrittelyt ja Sami Kojon vastuulla oli puolestaan tekninen toteutus.

Haluaisimme kiittää työn tarkastajaa professori Tommi Mikkosta erinomaisista kommenteista ja kärsivällisyydestä työn ohjauksessa. Lisäksi haluaisimme kiittää Jenni Laurilaa, Antti Ikäläistä ja Aappo Saloa oikoluvusta ja tärkeästä palautteesta.

Tampereella 23.2.2014

Arto Härkönen

Sami Kojo

SISÄLLYS

TERMIT JA NIIDEN MÄÄRITELMÄT

.NET	Microsoftin kehittämä sovelluskehys, joka tarjoaa yhteisen pohjan sitä käyttäville sovelluksille.
Action Mode	Ks. tehtävätila.
Affiinimuunnokset	Affiinimuunnokset ovat tietokonegrafiikassa yleisesti käytettäviä muunnoksia, joissa kappaleen pisteiden sijainnit eivät muutu suhteessa toisiinsa.
Atlas	Iso kuva, joka sisältää käyttöliittymään piirrettävät spritet.
CryEngine	Crytek-yrityksen kehittämä pelinkehitystyökalu.
DirectX	Microsoftin kehittämä tietokonegrafiikan piirtoon käytetty rajapinta.
Dispatch Mode	Hallinta- ja tehtävätilan välissä oleva tila, jossa tehtävään valitaan sopiva pelastuskokoonpano.
DLC	Peliin julkaisun jälkeen ilmaiseksi tai maksua vastaan internetin kautta hankittava lisäsisältö (Downloadable content).
FPS	Ruudunpäivitysnopeus. Kuvastaa, kuinka monta kertaa ruutu päivitetään sekunnissa (Frames per second).
Freeplay-pelimuoto	Rescue Squad -pelin toinen pääpelimuoto, jossa tehtäviä suoritetaan yhdellä pelikentällä ilman pidempiaikaisia tavoitteita.
GameObject	Kantaluokka, josta kaikki pelimaailman oliot periytyvät. GameObjectiin voi liittää MonoBehaviour-luokasta perittyjä olioita komponentteina.
GDD	Pelin suunnitteludokumentti, joka tyypillisesti sisältää kuvaukset pelin tärkeimmistä ominaisuuksista (Game Design Document).
Hallintatila	Kampanjapelimuodon tila, jossa pelaaja hallinnoi paloasemaa ja siihen kuuluvia elementtejä, kuten henkilökuntaa ja pelastusajoneuvoja.
IDE	Ohjelmointiympäristö, joka sisältää minimissään tekstieditorin ja kääntäjän (Integrated Development Environment).
Kulmapiste	Monikulmion kulman sijainnin määrittävä arvo.
Lokalisointi	Pelin kääntäminen käyttäjän kielelle. Lokalisointiin liittyy usein muutakin kuin tekstien kääntäminen. Esimerkiksi kuvista voi olla tarve luoda eri versiot eri kulttuuritaustan omaaville käyttäjille.

Management Mode	Ks. hallintatila.
Mesh	Ks. monikulmioverkko.
Monikulmio	Kolmesta tai useammasta kulmapisteestä muodostunut kuvio.
Monikulmioverkko	Monikulmioista muodostuva verkko. Monikulmioverkkoa käytetään kuvaamaan 3D-kappaleita.
Mono	Avoimen lähdekoodin toteutus Microsoftin .NET sovelluskehyksestä.
MonoBehaviour	Kantaluokka kaikille luokille, jotka käsittelevät pelissä olevia peliolioita.
OpenGL	Tietokonegrafiikan piirtoon käytetty avoin rajapinta.
Pelikonsepti	Korkean tason varhaisen vaiheen suunnitelma siitä, millainen pelin tulisi olla.
Pivot point	Monikulmioiden erikseen määriteltävä keskipiste, jonka ympäri monikulmion määrittelemä kappale pyörii 3D-avaruudessa.
Polygon	Ks. monikulmio.
Porttaus	Ohjelman muokkaaminen ja kääntäminen siten, että se toimii muulla kuin sillä alustalla, jolle se on alunperin kehitetty.
Prefab	Prefab on uudelleenkäytettävä GameObject, josta voidaan tehdä pelimaailmassa toimiva kopio kloonaamalla se pelitilaan.
Prototyyppi	Varhainen testiversio pelistä, jolla testataan peliin suunniteltuja ominaisuuksia.
Sandbox	Ks. Freeplay-pelimuoto.
Shader	Ks. sävytin.
Sprite	3D-näkymään sijoitettava bittikarttakuva.
Steam	Valve Corporationin kehittämä videopelien digitaalinen kauppapaikka.
Sävytin	Määrittelykieli, jolla voidaan laskea piirrettävän pinnan pisteen väri käyttäen parametreina esimerkiksi kappaleen väriä, valon väriä, valon tulokulmaa ja valon voimakkuutta.
Tehtävätila	Pelin tosiaikainen strategiatila, jossa pelaaja suorittaa annettuja tehtäviä.
Tekstuuri	Pinnalle piirrettävä kaksiulotteinen kuva.
UDK	Epic Games -yrityksen kehittämä pelinkehitystyökalu (Unreal Development Kit).

Unity	Rescue Squad -projektissa käytetty Unity Technologies -yrityksen kehittämä pelinkehitystyökalu.
UV-koordinaatti	Kaksiulotteinen kartta, jonka perusteella tekstuuri piirretään monikulmioverkon pinnalle.
Vertex	Ks. kulmapiste.

1 JOHDANTO

Tämä diplomityö käsittelee Fragment Production Oy:n tuottaman Rescue 2013: Everyday heroes -tietokonepelin kehitysprojektia. Projekti sai alkunsa, kun projektin tilaajana toiminut saksalainen pelijulkaisija Rondomedia GmbH kiinnostui eräästä yrityksen Rescue Squad -nimisestä pelikonseptista Saksassa järjestetyillä pelialan messuilla. Rondomedia ei kuitenkaan hyväksynyt pelikonseptia sellaisenaan, vaan heidän markkinointikoneistonsa halusi muokata projektia Saksassa hyvin menestyneitä simulaatiopelien suuntaan. Aluksi pelikonsepti oli tarkoitus kohdentaa mahdollisimman laajalle yleisölle. Tämä tarkoitti sitä, että graafinen tyyli ei ollut riittävän realistinen, eikä se näin ollen sopinut simulaatiopelien pelaajille. Kuvassa 1.1 on esitetty julkaisijan toiveen suuntaan muokattu konseptikuvitus Rescue Squad -pelistä.

Kuva 1.1 *Rescue Squad -projektin alustava konseptikuva (Rescue Squad -conceptual screenshot 2012).*

Samaan aikaan Rondomedia halusi myös vaihtaa suunnitellun kohdelaitteen mobiililaitteesta (lähinnä iPad ja toissijaisena iPhone) pöytätietokoneeseen Windows-alustalle

(myöhemmin myös Mac OSX). Varsinkin tämä muutos tarkoitti koko konseptin uudelleen kirjoittamista ja suunnittelua jo pelkästään ohjainlaitteiden fundamentaalisen muutoksen takia. Tässä siis ohjausrajapinta vaihtui mobiililaitteen kosketusnäytöstä pöytätietokoneella yleisempään ja paremmin toimivaan hiireen ja näppäimistöön.

Tässä diplomityössä käsitellään Rescue Squad -projektin taustoja, määrittelyjä ja teknistä toteutusta. Alun perin projekti oli tarkoitus toteuttaa kymmenessä kuukaudessa. Aikataulu oli jaettu viiteen välietappiin, jotka olivat kahden kuukauden päässä toisistaan. Projektin loppupuolella projektiin lisättiin kokonaan erillinen pelimuoto. Tämä pidensi projektin suunniteltua aikataulua noin 15 kuukauteen. Projektia laajennettiin ensimmäisen julkaisun jälkeen jälkituotantovaiheella, joka kesti joulukuun 2013 loppuun asti. Jälkituotantovaiheen aikana pelistä muokattiin uusia versioita, joilla tavoiteltiin Saksan ulkopuolisia markkinoita.

Rescue Squad toteutettiin käyttäen Unity-pelinkehitystyökalua. Unityn käyttöön päädyttiin erityisesti sen geneerisen luonteen vuoksi ja lisäksi, koska projektin henkilöstöllä oli aiempaa kokemusta Unityn käytöstä. Unityä ei siis ole jäykästi tarkoitettu millekään tietyille peligenreille, vaan se on suunniteltu mahdollisimman yleisluontoiseksi, jättäen pelikehittäjälle paljon vastuuta ja tämän myötä mahdollisuuksia laajentaa alustan toiminnallisuutta. Esimerkkinä vastakkaisesta suunnittelufilosofiasta on Unreal Development Kit (UDK), joka on pääsääntöisesti tarkoitettu first/third person -peleille. Lisäksi Unity tukee muitakin alustoja kuin Microsoft Windows -alustaa, mikä mahdollistaa myöhemmät mahdolliset käännökset muille alustoille.

Kaikki projektiin määritellyt välietapit saavutettiin onnistuneesti ajallaan. Osa välietappien määrittelyistä vaati kuitenkin pieniä iteratiivisia muutoksia. Ensimmäinen versio Rescue Squad -pelistä julkaistiin Saksassa pelikaupoissa myytävänä jälleenmyyntipakettina sekä digitaalisena www.simuwelt.de ja www.amazon.de-palveluiden kautta 2013 toukokuussa ja viimeisin amerikkalaisen Valve Corporationin hallinnoimassa Steam-latauspalvelussa 2013 joulukuussa. Steam on maailman suurin pelien digitaalinen kauppapaikka. Steamissa on yli 2000 peliä myynnissä (Steam 2014), ja yhtäaikaisia käyttäjiä on ollut kirjautuneena parhaimmillaan yli seitsemän miljoonaa (Pelaajalehti 2013). Valven palveluun pääsy oli suuri läpimurto sekä Fragment Productionin että Rondomedian kannalta, jolle Rescue Squad oli ensimmäinen kyseiselle digitaaliselle julkaisualustalle päätynyt tietokonepeli.

Luvussa kaksi käsitellään diplomityön ymmärtämisen kannalta tarpeellisia taustoja. Luvussa esitellään muun muassa tietokonegrafiikan perusteet, joiden ymmärtäminen on tärkeää työssä myöhemmin esiteltävien teknisten ongelmien vuoksi. Lisäksi luvussa esitellään Rescue Squad -projektin taustaa. Luku kolme käsittelee projektin vaatimuksia. Luvussa neljä esitellään projektin toteutuminen ja suurimmat tekniset haasteet. Luvussa viisi arvioidaan projektin kokonaisvaltaista onnistumista ja luvussa kuusi kootaan diplomityö yhteen.

2 TAUSTAA

Tässä luvussa kuvataan peliprojektien ymmärtämisen kannalta välttämätöntä taustaa sekä Rescue Squad -projektin taustoja. Kohta 2.1 antaa yleiskuvan tietokonegrafiikasta, joka on yksi tärkeimmistä tietokonepelien osa-alueista. Kohta 2.2 taustoittaa, millaisen henkilöstön tyypillinen pelinkehitysprosessi vaatii ja kuvaa, millainen henkilöstö toteutti Rescue Squad projektin. Kohdassa 2.3 kuvaillaan yleisempiä peligenrejä. Rescue Squad -projektin kannalta tämä luku on erityisen merkityksellinen, koska projektin lopputuloksena syntynyt peli ei sovi suoraan yhteenkään yksittäiseen genreen.

Kohta 2.4 kuvaa tyypillisen peliprojektin läpivientiä ja kuinka projekti vietiin läpi Rescue Squadin tapauksessa. Kohta 2.5 kuvaa Unity Technologies -yrityksen kehittämää Unity pelinkehitystyökalua. Tämä kohta on erityisen tärkeä Rescue Squad -projektin kannalta, koska projekti toteutettiin käyttäen Unityä lopputuloksena syntyneelle pelille.

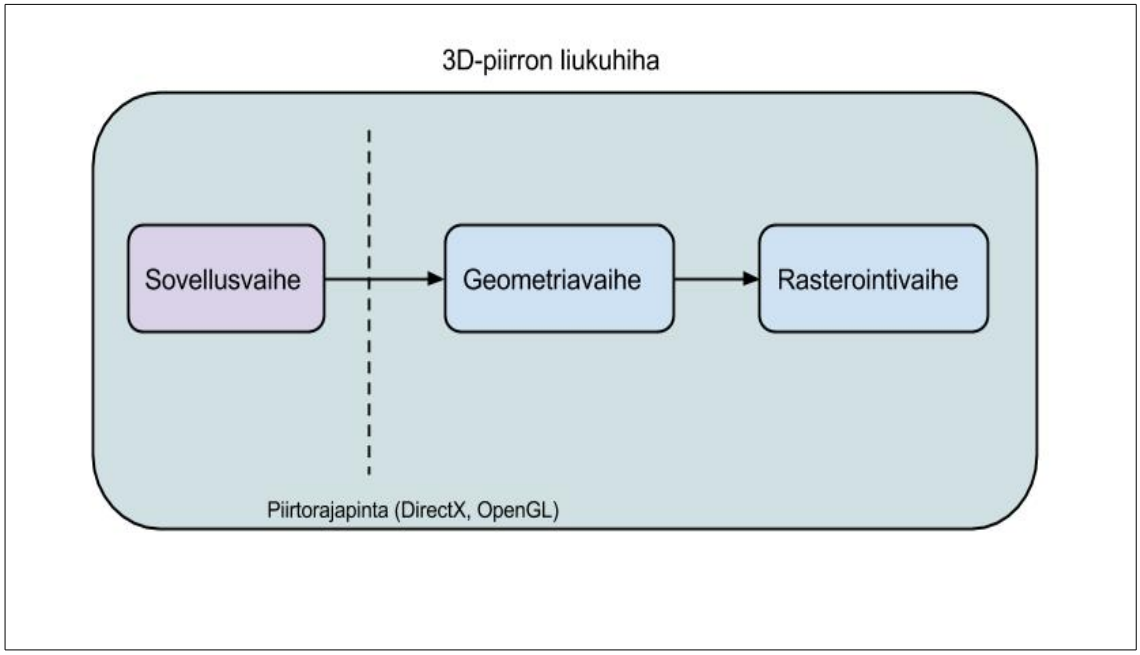
2.1 Tietokonegrafiikka

Tietokonepeleissä pääosassa on esitetty grafiikka. Grafiikkaa voidaan tuottaa monella tavalla, mutta Rescue Squad -projektissa on käytetty vain 3D-pohjaista grafiikkaa. 3D-grafiikan piirto jakautuu kolmeen päävaiheeseen, jotka on esitetty kuvassa 2.1.

Näistä ensimmäisessä, sovellusvaiheessa, tietokonepelin kehittäjä määrittelee, mitä myöhemmissä vaiheessa piirretään. Tämä tapahtuu määrittelemällä kolmiulotteisia malleja, jotka koostuvat kulmapisteistä (vertex) sekä näiden avulla muodostetuista viivoista ja monikulmioista. Tuloksena syntyneet mallit lähetetään piirtorajapinnan kautta myöhemmille vaiheille, joista ensimmäinen on geometriavaihe. Geometriavaihe suoritetaan nykyaikaisissa tietokonejärjestelmissä grafiikkaohjaimella (GPU), kun taas sovellusvaihe on pääosin keskusyksikön (CPU) hoidettavana. Koska keskusyksikkö ja grafiikkaohjain ovat tyypillisesti erillisiä prosessointiyksiköitään, tietopaketit näiden välillä täytyy siirtää yksiköiden välillä olevaa väylää pitkin. Kun piirrettävää informaatiota on paljon eli siirrettävä tietomäärä kasvaa suureksi, tulee kyseinen väylä usein piirtonopeutta rajoittavaksi tekijäksi. Tämä täytyykin ottaa huomioon piirtonopeuden optimoimisessa jo sovellusvaiheessa. (Puhakka 2008, s. 164-165).

Geometriavaiheen tehtävä on muuntaa sovellusvaiheen antama kolmiulotteinen malli rasteroitavaksi kolmannessa ja viimeisessä rasterointivaiheessa. Tämä tehdään käyttämällä geometrisia muunnoksia (affiinimuunnokset), joilla 3D-avaruuden pisteet lopulta projisoidaan 2D-pinnalle. Nykyaikaisissa järjestelmissä geometriavaihe voidaan toteut-

taa kuhunkin tarkoitukseen erikseen ohjelmoitavalla verteksisävyttimellä (vertex shader), kun taas aiemmin tämä vaihe oli kiinteästi toteutettu grafiikkaohjaimeen. (Puhakka 2008, s. 166-169).



Kuva 2.1 *3D-piirron kolme eri vaihetta (Puhakka 2008, s 164).*

Viimeiselle eli rasterointivaiheelle jää tehtäväksi piirtää lopullinen käyttäjälle näkyvä rasterikuva. Tämä tapahtuu muuntamalla geometriavaiheen tuottama tieto pikseliruudukolle sopivaan muotoon ja laskemalla jokaiselle piirretylle mallille ja sen määrittämälle pikselijoukolle lopullinen värinsä käyttäen pikselisävytintä (fragment/pixel shader). Kuten verteksisävyttimen kohdalla, myös pikselisävytin voidaan ohjelmoida tuottamaan erikoistehosteita kiinteän ohjelman sijaan. Tyypillisesti pikselin väri määräytyy mallitietoon sisälletyn tekstuurin avulla, mutta monimutkaisemmissa toteutuksissa väriin voi vaikuttaa myös muut 3D-maailmassa sijaitsevat asiat, kuten valonlähteet. (Puhakka 2008, s. 170-171).

Nykyisin 3D-grafiikan piirtoon käytetään pääasiassa kahta keskenään kilpailevaa rajapintaa. OpenGL on avoin ja alustariippumaton, kun taas Microsoftin kehittämä DirectX on suljettu rajapinta. OpenGL-standardin määrittelee OpenGL Architecture Review Board (ARB), johon kuuluvat useat suuret tietokonegrafiikan parissa olevat toimijat. OpenGL:n ehdottomiin vahvuuksiin kuuluu sen alustariippumattomuus, mikä tarkoittaa, että siitä on toteutuksia Windows-käyttöjärjestelmän lisäksi myös Linux:lle sekä Mac OSX:lle (Puhakka 2008, s 355). Myös nykyaikaiset mobiilialustat, kuten Android (Android developers 2013) ja Applen iOS (iOS Developer Library 2013), tukevat OpenGL:ää.

Toisin kuin OpenGL, Microsoftin DirectX on sidottu Windows-käyttöjärjestelmään. Tästä huolimatta se on saavuttanut merkittävän aseman tietokonepelien grafiikkaraja-

pintana (Puhakka 2008, s 355). Esimerkiksi Unity-pelinkehitystyökalu käyttää oletusarvoisesti DirectX-rajapintaa Windows-tietokoneilla, vaikka OpenGL-tuki Windows-tietokoneille on myös toteutettuna. Käyttäjä saa kuitenkin halutessaan OpenGL-rajapinnan käyttöön, mutta se vaatii erillisen komentoriviparametrin Unityä käynnistettäessä (Unity 2013a).

Rescue Squad -projektin kehityksen kannalta oleellisin vaihe on sovellusvaihe, jossa luodaan piirrettävät mallit ja määritellään niiden piirtämiseen tarkoitetut sävyttimet. Rescue Squad -projektissa tässä apuna oli Unity-pelinkehitystyökalu, joka toimii ikään kuin erillisenä vaiheena sovellusvaiheen ja piirtorajapinnan välissä ollen kuitenkin osa sovellusvaihetta. Unity siis tarjoaa joukon valmiita ratkaisuja kehitystyön nopeuttamiseksi ja helpottamiseksi.

2.2 Pelinkehityshenkilöstö

Peliprojektin olennaisin resurssi on pelinkehitykseen osallistuvat tekijät. Tämä tarkoittaa sitä, että alkuun päästäkseen pelinkehitysyrityksen ei tarvitse aluksi tehdä kattavia investointeja raskaisiin laitteisiin tai muuhun yritysinfraan, vaan alkuun pääsee jo muutaman tuhannen euron sijoituksella pöytätietokoneisiin ja mahdollisiin ohjelmistolisensseihin. Alunkin jälkeen lähes kaikki kulut koostuvat henkilöstökuluista eli palkoista ja näiden sivukuluista.

Muuhun it-teollisuuteen nähden pelikehitystiimit ovat yleisesti ottaen varsin pieniä, ja pienimmillään jopa yksi henkilö voi saada menestyvän pelin toteutettua ja julkaistua. Yksi tällainen esimerkki on oululainen Fingersoft Oy, joka julkaisi vuonna 2012 vain yhden tekijän työpanostuksella tuotetun Hill Climb Racing -mobiilipelin. 17.12.2013 mennessä peliä on ladattu eri mobiilialustoilla jo yli 100 miljoonaa kertaa (Hill Climb Racing 2013).

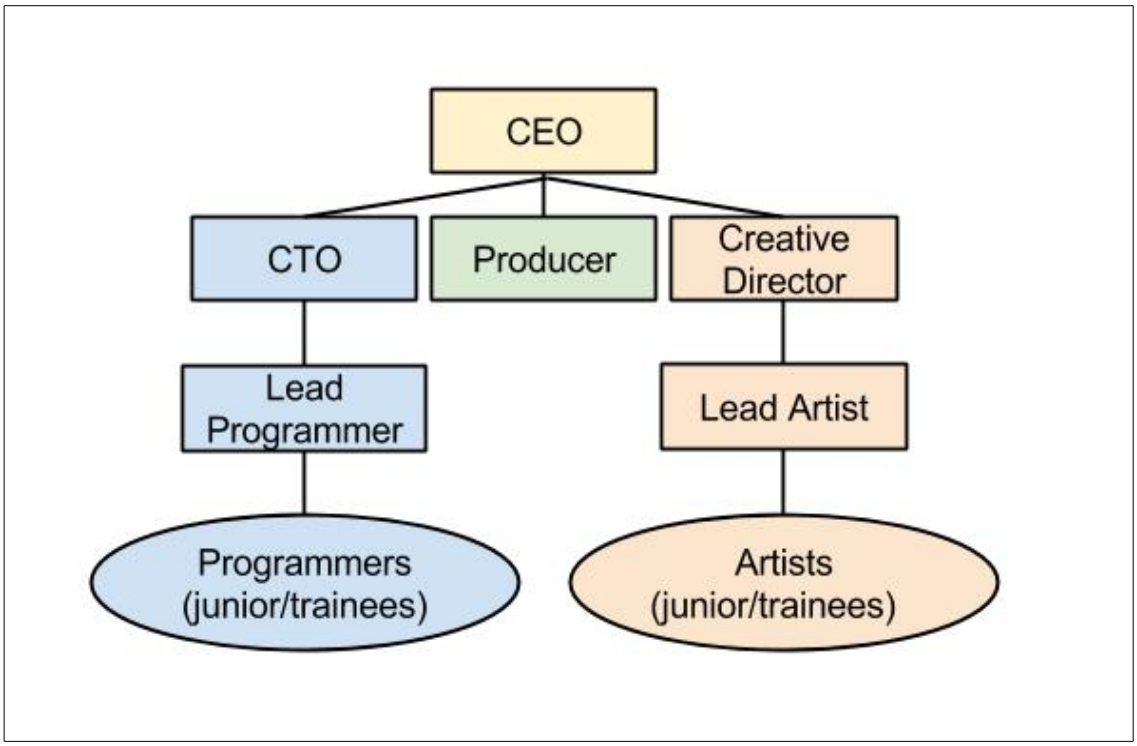
Tiimit koostuvat pääasiallisesti kolmesta pääroolista: ohjelmoijat, suunnittelijat ja graafikot (Flynt 2005, p 474). Kuva 2.2 esittää Rescue Squad -projektin tuottaneen Fragment Production Oy:n sisäistä hierarkiaa. Huomattavaa on, että Rescue Squad -projektin aikana ei ollut meneillään muita projekteja, joten yrityksen koko henkilöstö osallistui Rescue Squad -projektin toteuttamiseen.

2.3 Peligenret

Eri tyyppiset pelit jaetaan peligenreihin, jostain yleisempiä on esitelty taulukossa 2.1. Näiden genrejen lisäksi on runsaasti genrekohtaisia aligenrejä sekä yhdistelmiä näistä päägenreistä. Esimerkiksi Rescue Squad -projekti on yhdistelmä simulaatio-, roolipeli- ja strategiagenrejä, eli siitä on löydettävissä näiden genrejen tyypillisimpiä ominaisuuksia.

Rescue Squad on ensisijaisesti tosiaikainen strategiapeli. Tämä tarkoittaa sitä, että yksi sekunti pelimaailmassa vastaa yhtä sekuntia tosimaailmassa. Tosiaikaisille strate-

giapeleille on tyypillistä, että pelin osapuolet suorittavat toimintojaan yhtä aikaa. Vastakohtana ovat vuoropohjaiset strategiapelit, joissa pelaajat toimivat vuorotellen. Tosiaikaisissa strategiapeleissä vastakkain voi pelata yksi tai useampi ihmisen tai tietokoneen komentama joukko yksiköitä (Walker 2003, p 10). Rescue Squad on yksin pelattava peli, jossa vastustajana toimii palosimulaatio ja aika.



Kuva 2.2 *Fragment Production Oy:n Rescue Squad -projektin aikainen organisaatiokaavio.*

2.4 Peliprojektin läpivienti

Tavanomainen tapa pilkkoa peliprojekti on osittaa se välietappeihin (milestone). Näistä tyypillisimmät ovat alpha, beta ja gold master (Flynt 2005, p 172). Taulukossa 2.2 esitetään välietappien esimerkkitavoitteet.

Monessa tapauksessa Gold Master -versioonkin jää vielä virheitä, jotka korjataan varsinaisen julkaisun jälkeen tehdyillä korjauspaketeilla. Julkaisijoiden kanssa toimittaessa on tavallista, että julkaisusopimus pitää sisällään tietyn aikajakson, jolloin kehittäjä on velvollinen korjaamaan käyttäjien pelistä löytämiä virheitä.

Alakohdassa 2.4.1 käydään läpi, miten Rescue Squad -projektia varten haettiin tarpeelliset tiedot tavoitteiden määrittelemiseksi. Alakohta 2.4.2 kuvaa, kuinka peliprojektin tavoitteet tyypillisesti määritellään, ja kuinka tämä on toteutettu Rescue Squad -projektissa. Alakohdassa 2.4.3 kuvataan peliprojektin toteutuksesta yleisesti ja Rescue Squad -projektin näkökulmasta. Projektin päättäminen kuvataan alakohdassa 2.4.4.

Taulukko 2.1 *Yleisimmät peligenret (Walker 2003, pp. 9-13).*

Genre	Kuvaus	Esimerkkipelejä
Toiminta	Toimintapelit perustuvat pelaajan henkilökohtaiseen taitoon eli yleisesti ovat pelejä, joissa pelaajan reflekseillä on suuri merkitys pelissä pärjäämiseen.	Unreal Tournament, Quake-sarja, Call of Duty -sarja.
Strategia	Toimintapelejä huomattavasti hidastempoisempia, jossa pelaajan päätöksillä ja valinnoilla voi olla kauaskantoisia vaikutuksia.	Transport Tycoon, Theme Hospital, Command and Conquer -sarja.
Roolipelit	Roolipeleissä pelaajan fokuksessa on pelihahmon kehittäminen, joka tapahtuu yleisesti keräämällä kokemuspisteitä. Kokemuspisteitä puolestaan kertyy suorittamalla ei-pelaajahahmojen antamia tehtäviä tai tappamalla maailmassa olevia vihollisia.	Baldurs Gate -sarja, Fallout-sarja.
Simulaatiot	Simulaatiopelit keskittyvät mallintamaan mahdollisimman tarkasti jotain aspektia oikeasta maailmasta. Tällainen voi olla esimerkiksi jokin ammattikunta ja heidän päivittäiset tehtävänsä.	Flight Simulator -sarja, America's Army.
Pulmapelit	Pelaajalle annetaan ratkaistavaksi jonkinlainen haastava ongelma. Yleensä pulman ratkaisusta saadaan palkinnoksi pisteitä, joiden kautta voidaan kilpailla muiden pelaajien kanssa.	Tetris, Puzzle Quest, Bejeweled.

2.4.1 Taustatyö

Pelikokemuksen immersion kannalta on arvokasta, että pelissä tehdyt viittaukset oikeaan maailmaan ovat mahdollisimman realistiset. Viittausten oikeellisuus on erityisen tärkeää simulaatiopeleissä, jotka pyrkivät mallintamaan reaalimaailman ilmiöitä mahdollisimman tarkasti. Rescue Squad -projektin alkuvaiheessa olikin tarpeen tutustua tarkemmin sammutusvälineistöön ja paloasemalla henkilöstön keskuudessa vallitsevaan kulttuuriin pelin hallintatilaa silmällä pitäen. Kirjallisuudesta saatuja tietoja täsmennettiin muun muassa paloautojen välineistön ja niiden kuljetuskapasiteetin osalta. Oli myös tärkeää saada tuntumaa siitä, miten suojapuvut vaikuttavat liikkuvuuteen ja näin saada paremmin käsitystä niiden käyttäytymisestä esimerkiksi animaatioiden tekemistä varten.

Kuvassa 2.3 on Rescue Squad -projektihenkilöstöä kokeilemassa savusukelluspukua Tampereen VPK:n harjoituksissa.

Taulukko 2.2 *Välietapit ja niiden tuotantotavoitteet (Flynt 2005, p 172).*

Välietappi	Kuvaus
Ensimmäinen pelattava versio (proof of concept, first playable)	Esittelee pelin ydinominaisuuksia. Ei juurikaan varsinaista pelisisältöä tai minkäänlaisia valmiita resursseja (assetteja).
Alpha	Tavoitteena esitellä esimerkkitapaukset kaikista pelin tärkeimmistä ominaisuuksista (featureista). Voi sisältää vielä virheitä.
Beta	Mahdollisimman virheetön, mutta hiomaton esitys pelistä. Alpha-versiossa olleet virheet on korjattu. Tavoitteena tuottaa peli, jonka voi jo pelata läpi ja peli on viihdyttävä.
Gold Master	Virheetön julkaisukelpoinen versio. Tiedostoista tehdään asennettava paketti, joka lähetetään esimerkiksi painettavaksi optiselle medialle.

2.4.2 Tavoitteiden määrittely

Pelin tavoitteet ja ydinominaisuudet kootaan alalla yleiseen tapaan niin kutsuttuun pelisuunnitteludokumenttiin (Game Design Document, GDD). Dokumentin laajuus vaihtelee eri yritysten käytännöistä riippuen suuresti muutaman sivun vision kiteytyksistä useampi satasivuisiin järkäleisiin, jotka pyrkivät kuvaamaan koko pelin pienintäkin yksityiskohtaa myöten. Rescue Squad -projektissa noudatettiin alun perin jälkimmäistä mallia eli ensimmäinen versio pelisuunnitteludokumentista oli laajuudeltaan noin 80 sivua pelin yksityiskohtiin menevää määrittelyä. Toteutusvaiheessa tätä mallia kuitenkin iteroitiin paljon, ja raskaan pelisuunnitteludokumentin sijaan siirryttiin erillisiin ominaisuusmäärittelyihin, jotka olivat korkeintaan muutaman sivun pituisia. Jokaisesta uudesta ominaisuudesta tehtiin sarjakuvamaiset kuvaukset, joista kävi helposti ilmi pelaajan suorittamat tehtävät ja näihin pelin puolelta annetut vasteet.

2.4.3 Toteutus

Kun projektin alkuvaiheen tavoitteet on määritelty ja lyöty lukkoon, voidaan aloittaa pelin varsinainen tuotanto. Muuhun it-teollisuuteen verrattuna tietokonepelien toteutus noudattaa yleisesti hyvin vahvaa iteratiivista linjaa, eikä siis kovinkaan hyvin taivu jäykkään vesiputousmalliin. Syy tähän on se, että ennen varsinaista toteutusta pelin toimivuutta on äärimmäisen vaikea arvioida ennalta. Siksi ensimmäisessä vaiheessa on tärkeää tehdä pelin ydinominaisuuksia testaava prototyyppi, jonka perusteella kehittäjä saa paremman arvion ominaisuuksien toimivuudesta. Rescue Squadin tapauksessa tulipalo-

jen sammutuksesta ja hahmojen sekä kulkuneuvojen hallinnasta saatiin prototyypin avulla tärkeää tietoa. Tosin aikataulullisista syistä tämä vaihe jäi liian vähälle huomiolle, ja tästä seurasikin pelillisiä ja teknisiä ongelmia, jotka seurasivat läpi koko projektin. Näistä ehdottomasti merkittävin oli teknisten haasteiden kasaantuminen pelin reitinhakujärjestelmään ja sen liialliseen monimutkaistumiseen.

Kuva 2.3 *Rescue Squad -projektin henkilöstöä tekemässä taustatutkimusta Tampereen VPK:n harjoituksissa.*

Prototyyppivaiheen aikana voidaankin vielä tehdä suuriakin muutoksia määrittelyihin, jos todetaan, että alkuperäinen suunnitelma oli virheellinen, tai jopa mahdollisesti mahdoton toteuttaa teknisesti tai projektin aikataululle sopivasti. Kyseinen vaihe on siis hyvin kriittinen koko projektin onnistumisen kannalta eikä näin ollen vaiheessa ole suositeltavaa kiirehtiä. Prototyyppivaihe päättyy ensimmäisen pelattavan version valmistumiseen, joka on yleensä välietappimäärittelyssä projektin ensimmäinen varsinainen määräaika. Rescue Squad -projektin kohdalla tämä vaihe kesti noin puolitoista kuukautta.

Tämän jälkeen peliä kehitetään suunnitellussa aikataulussa välietappeineen kohti Gold Master -julkaisua, jolloin peliprojektin varsinainen toteutusvaihe päättyy ja siirrytään niin kutsuttuun jälkituotantovaiheeseen. Tällöin voidaan vielä poistaa korjauspäivityksillä peliin mahdollisesti jääneitä tai julkaisun jälkeen havaittuja virheitä. Lisäksi kielikäännökset kuuluvat yleisesti vasta tähän vaiheeseen.

Rescue Squad -projektin kohdalla laadunvarmistus alihankittiin Rondomedian toimesta saksalaiselta Games Quality -yritykseltä, joka teki myös varsinaisen virhe-etsinnän lisäksi arvioita pelin pelattavuudesta sekä tutki, onko tuote pelillisesti viihdyttävä ja eheä kokonaisuus. Arvio sisältää myös analyysin pelin soveltuvuudesta ennustetulle kohdeyleisölle. Yritys hankkii kohdeyleisöön sopivia henkilöitä, joilla he peluuttavat analysoitavaa peliä, ja tekee johtopäätöksiä havaintojensa perusteella.

2.4.4 Projektin päättäminen ja korjauspäivitykset

Onnistunut peliprojekti päättyy pelin Gold Master -välietappiin, jolloin julkaistavaksi lähetetään kaupalliseen levitykseen menevä levykuva, joka sisältää pelistä tehdyn asennuspaketin. Digitaalisessa julkaisussa erillistä levykuvaa ei tarvita, vaan asennuspaketin voi julkaista sellaisenaan, riippuen tietysti julkaisualustan luonteesta. Esimerkiksi Steam-latauspalvelun kohdalla alusta itse hoitaa myös pelin asentamisen, jolloin käyttäjän ei tarvitse maksun jälkeen kuin valita asenna-vaihtoehto ja odottaa pelin asentumista taustalla.

Peliprojektit eivät julkaistuinakaan ole täysin valmiita, vaan lähes aina voidaan löytää korjattavaa ja parannettavaa. Tämä johtuu pelien ominaispiirteestä, jossa yhdistyy tekninen ja luova osaaminen. Varsinaisten virheiden lisäksi peli ei välttämättä ole täysin tasapainossa pelillisesti, vaan pelin kehittäjien aluksi määräämiä vakioita saatetaan joutua muuttamaan paremman pelielämyksen tuottamiseksi. Tämä on seurausta peliprojektien loppupuolella tapahtuvasta ilmiöstä, jossa kehittäjistä itsestään tulee liian taitavia pelaamaan omaa peliään. Nyt uuden pelaajan näkökulmasta pelistä tulee liian haastava aiheuttaen turhautumista, koska eteneminen on liian työlästä tai haastavaa. Riski tähän ilmiöön on erityisen korkealla, jos tuotannon loppupuolella ei käytetä ammattimaista laadunvarmistukseen erikoistunutta henkilöstöä.

Yllä kuvatuista syistä on tavallista, että pelejä korjauspäivitetään julkaisun jälkeen, jotta edellä mainitut epätasapaino-ongelmat poistuisivat. Huomion arvoista on myös, että tätä ei ole aina voitu tehdä, kun pelejä on poltettu esimerkiksi kerrankirjoitettaville ROM-piireille. Koska internetin yleistyminen on mahdollistanut korjauspäivittämisen, pelit julkaistaan hyvin usein varsin keskeneräisinä ja näin ollen ne vaativat julkaisun jälkeistä korjailua.

Korjauspäivitysten lisäksi nykypeleille tyypillinen jälkituotantoprojekti on tuottaa kaupallisesti onnistuneesta projektista erikseen digitaalisesti ladattavia lisäosia. Näillä yritetään saada pelistä kaikki kaupallinen hyöty irti. Myytäviä lisäosia voisi olla esimer-

kiksi uudet karttapaketit. Rescue Squadin tapauksessa jälkituotanto keskittyi uusiin paloautoihin, tehtäviin sekä hahmoihin erillisten maaversioiden muodossa.

2.5 Unity-pelinkehitystyökalu

Rescue Squad toteutettiin käyttäen Unity Technologies -yrityksen kehittämää Unity-pelinkehitystyökalua. Unity tarjoaa pelimoottorin lisäksi graafisen editorin pelikenttien luomiseen. Unityn mukana toimitetaan myös avoimen lähdekoodin IDE MonoDevelop, jota voi lähdekoodin kirjoittamisen lisäksi käyttää virheiden löytämiseen Unity-pelien lähdekoodista.

Tärkein syy Unityn valintaan oli Unityllä tehtyjen projektien helppo siirrettävyys alustalta toiselle. Unity tukee niin Microsoftin Windows, Applen OSX ja Linux käyttöjärjestelmiä pöytätietokoneiden puolella. Mobiililaitteille tarkoitetuista käyttöjärjestelmistä Unity tukee Applen iOS:ää, Googlen Androidia, Microsoftin Windows Phonea (versiosta 8 lähtien, tuki tälle lisättiin Unityn versioon 4.2) ja BlackBerry Ltd:n (entinen RIM, Research In Motion) BlackBerry 10:tä. Lisäksi Unityllä on mahdollista kehittää pelejä useille eri pelikonsoleille. Näiden tapauksessa lisensointiehdot on neuvoteltava Unity Technologiesin kanssa tapauskohtaisesti (Unity 2014c).

Unityn lisenssiehdot soveltuvat aloittelevalle yritykselle hyvin. Perusversio Unitystä on saatavilla ilmaiseksi. Tämä versio tarjoaa tuen kaikille niille pöytätietokoneiden ja mobiililaitteiden käyttöjärjestelmille, joita maksullinen versio tukee. Ilmaisessa versiossa on kuitenkin vähemmän ominaisuuksia kuin maksullisessa versiossa ja osa näistä ominaisuuksista, kuten suorituksen aikana luotavat navigaatioverkon esteet, olivat pakollisia pelin toteuttamiseksi. Lisäksi yritykset, joiden liikevaihto on enemmän kuin 100 000 yhdysvaltain dollaria, eivät voi lisenssiehtojen mukaan käyttää ilmaista versiota Unitystä (Unity 2014a). Näiden syiden vuoksi projektissa päädyttiin käyttämään maksullista versiota.

Unity tarjoaa sekä ilmaisessa että maksullisessa versiossaan tuen reitinhaulle. Unityn toteutus reitinhausta perustuu ennen ohjelman julkaisuversion luomista tehdystä navigaatioverkosta ja agenteista, jotka liikkuvat navigaatioverkkoa pitkin. Tarkemmin Unityn toteuttama reitinhakujärjestelmä on kuvattu alakohdassa 4.4.2.

Unityyn lisättiin monipuolinen Mechanim-animaatiojärjestelmä versiossa 4.0 (Unity 2013b). Muun muassa Mechanim tarjoaa animaatioiden toteuttamiseen graafisen tilakoneen, jonka avulla muutkin kuin ohjelmoijat voivat toteuttaa animaatiot.

Tehokkuutta parantamaan Unityn maksullinen versio tarjoaa työkalun näkyvyyskarsinnan toteuttamiselle (Unity 2013c). Näkyvyyskarsinnan tarkoitus on tehokkaasti välttää niiden mallin osien käsittely, joita pelaaja ei näe (Puhakka 2008, s. 259-260). Näkyvyyskarsintaa ei Rescue Squad -projektissa kuitenkaan käytetty, koska sillä saatava hyöty ei pöytätietokoneelle peliä tehdessä ole riittävän suuri vaadittavaan työmäärään nähden. Karsinnan tarvetta vähensi lisäksi Rescue Squadissa käytetty kuvakulma, jossa pe-

lialueesta nähdään muutenkin vain pieni pala kerrallaan, joten piirrettävä alue jää kentän kokoon suhteutettuna hyvin pieneksi.

Unityn muina vaihtoehtoina analysoitavana olivat Epic Games - yhtiön Unreal Developmet Kit sekä CryTek:n CryEngine, mutta molemmat todettiin nopeasti sopimattomiksi Rescue Squad -projektiin. Ensisijainen ongelma molempien kohdalla oli se, että ne ovat pääasiallisesti suunniteltu ensimmäisen persoonan kuvakulman toimintapeleihin, jollainen Rescue Squadin ei siis pitänyt olla. Vastaavia tosiaikaisia strategiapelejä ei analyysia tehdessä löytynyt kummankaan pelikehitysalustan varaan toteutettuna. Aiheesta löytyi vain prototyypin asteelle vietyjä projekteja, mutta ei mitään niin kehittynyttä, että UDK:n tai CryEnginen käyttöönotto olisi ollut järkevää Rescue Squad -projektin kannalta.

Sekä UDK:ssä että CryEnginessä oli lisäksi Unitystä poikkeavat lisensointimallit, joissa tietyn alarajan ylittävästä tuotosta täytyy maksaa lisenssimaksua. UDK:n kohdalla tämä prosentti oli 25% (UDK 2014) ja CryEnginen 20% (CryEngine 2014). Unityyn verrattuna nämä vaihtoehdot olisivat Rescue Squad -projektissa aiheuttaneet hieman enemmän kustannuksia, mutta eivät kuitenkaan niin merkittävästi, että kustannusvaikutuksella olisi ollut merkitystä pelinkehitystyökalun valinnan kannalta.

3 RESCUE SQUAD -PROJEKTIN VAATIMUKSET JA MÄÄRITTELYT

Luvussa 3 kuvataan Rescue Squad -projektin vaatimukset ja tekniset määrittelyt. Kohta 3.1 kuvaa pelin sisällön ja selventää, miten peli jakautuu erilaisiin pelitiloihin. Kohdassa 3.2 kuvataan pelin keskeisimpien ominaisuuksien vaatimukset. Kohdassa 3.3 kuvataan pelistä julkaistavat versiot. Kohta 3.4 kuvaa, miten projekti jaettiin välietappeihin ja mitä ominaisuuksia kunkin etapin lopputuloksen piti sisältää. Kohdassa 3.5 eritellään lopullisen pelin järjestelmävaatimukset ja kuvataan, miten näihin vaatimuksiin päädyttiin.

3.1 Pelin kuvaus

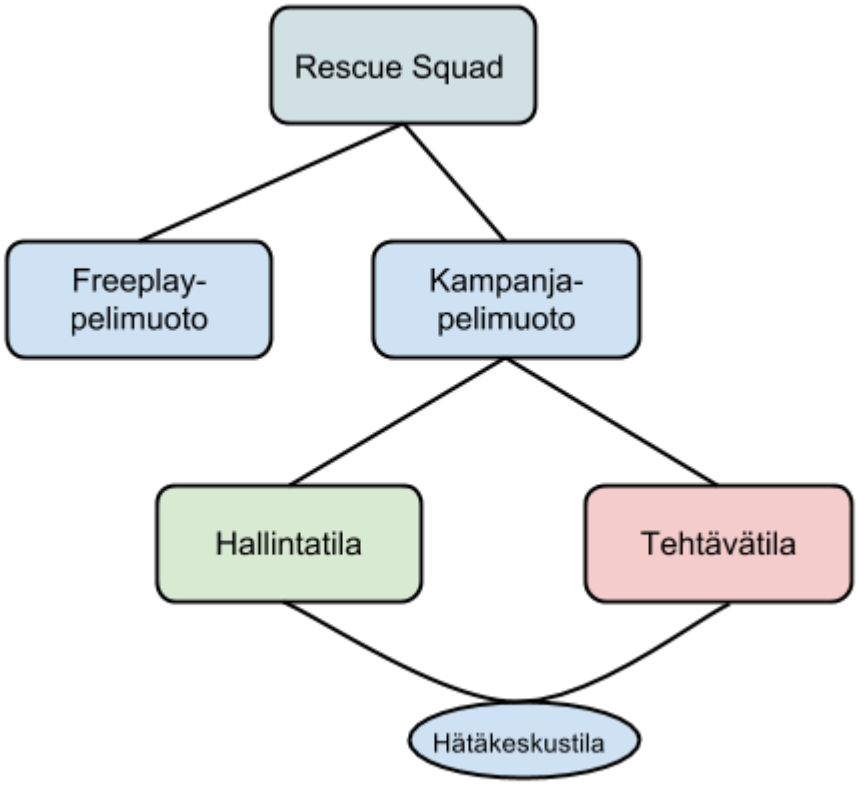
Pelin jakautuminen erilaisiin pääpelitiloihin on esitetty kuvassa 3.1. Kuvassa olevat kampanja- ja freeplay-pelimuodot (sandbox) on toisistaan riippumattomia pelikokonaisuuksia, jotka toimivat omilla säännöstöillään. Näistä huomattavasti laajempi on kampanjatila, joka edelleen jakautuu hallinta- ja tehtävätilaksi (Action mode, Management mode) kuvan 3.1 osoittamalla tavalla. Näiden lisäksi pelissä on lukuisia tuki- ja latausstiloja pelikokemuksen eheyttämiseksi.

Alakohdassa 3.1.1 esitellään vaatimukset pelin kampanjapelimuodolle, joka on pelin toinen pääpelimuodoista. Freeplay-pelimuoto puolestaan esitellään alakohdassa 3.1.2. Lopuksi alakohta 3.1.3 kuvaa tarpeet pelin erilaisille tukitiloille.

3.1.1 Kampanjapelimuoto

Pelin toisena pääasiallisena pelimuotona on roolipelielementtejä sisältävä kampanjapelimuoto, jossa pelaaja ottaa hoitaakseen paloaseman ja siihen liittyvän henkilöstön sekä pelastusajoneuvot. Pelaajan tehtävä on palkata ja kouluttaa henkilöstöä sekä hankkia uusia sammutuskalusteita samalla huolehtien, että paloaseman budjetti ei ylity. Resursseja pelaaja saa joka kuukausi tietyn summan, joka kasvaa pelin edetessä ja pelaajan maineen parantuessa. Lisäksi pelaaja saa pienen rahallisen bonuksen tehtävien suorittamisesta.

Realismin vuoksi paloasematoiminta ei sinänsä voi koskaan mennä konkurssiin, eli kampanjapelimuodossa ei ole lopetussääntöä. Huonolla pelityylillä eteneminen voi kuitenkin muodostua mahdottomaksi tai äärimmäisen vaikeaksi.

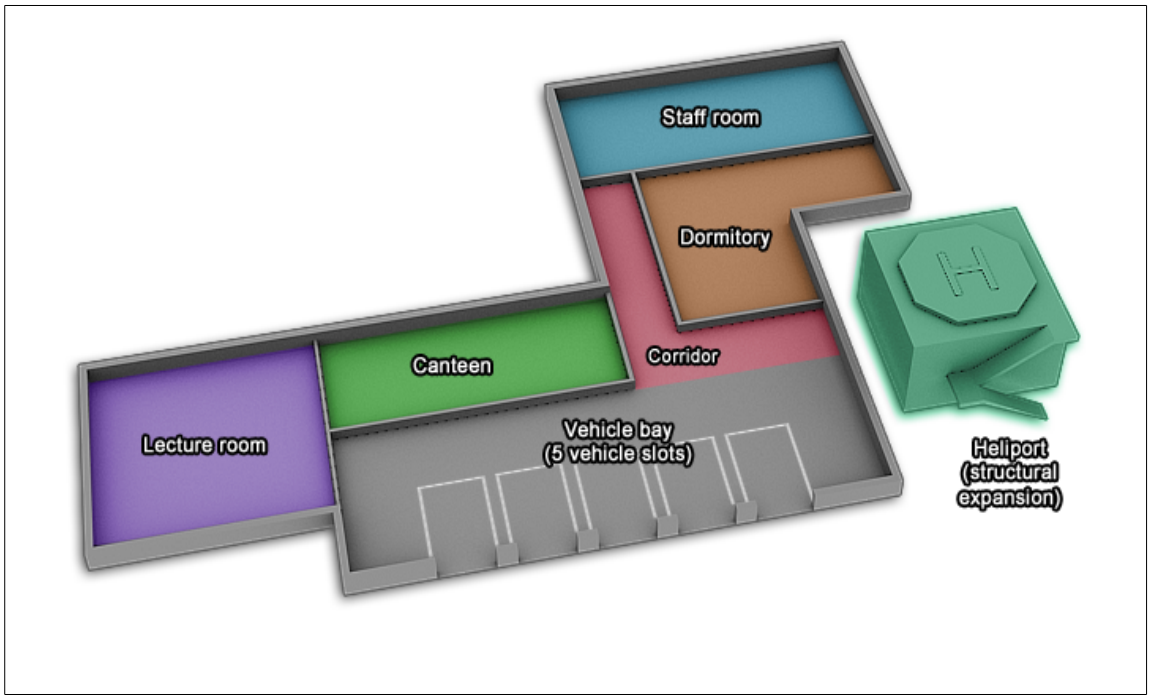


Kuva 3.1 Pelin jakautuminen kaaviona.

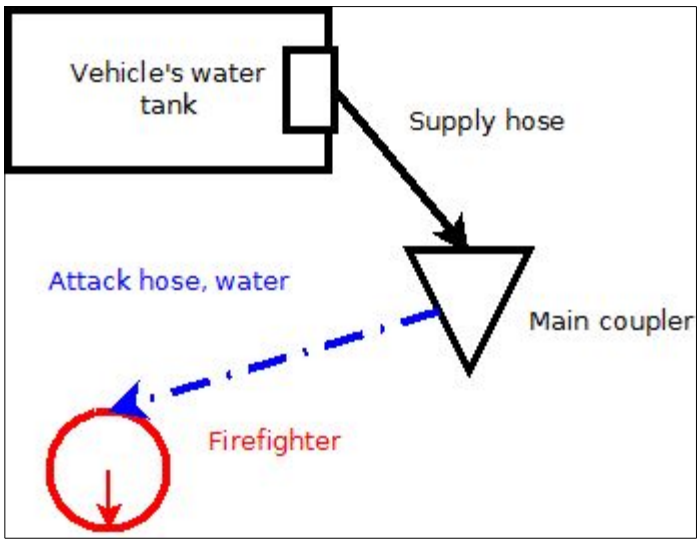
Hallintatila. Kampanjan pääasiallinen pelitila on hallintatila, jossa tehdään kaikki strategisen tason päätökset, kuten päätös uuden paloauton hankkimisesta. Operatiivisella tasolla pelaaja pitää huolen siitä, että henkilöstö ja kalusto on valmiina pelaajalle esitettäviin pelastustehtäviin. Henkilöstön koulutuksen lisäksi pelaajan tehtäviin kuuluu pitää asemalla töissä olevien palomiesten toimintakyky mahdollisimman korkealla. Toimintakykyä, mitataan moraaliarvolla, johon pelaaja voi vaikuttaa kalustamalla paloasemaa uusilla huonekaluilla ja viihdelaitteilla. Esimerkiksi biljardipöytä parantaa aseman henkilöstön viihtyvyyttä ja siten toimintakykyä. Kuvassa 3.2 on esitelty konseptuaalinen kuvaus paloasemasta ja sen sisältämistä huoneista.

Tehtävätila. Varsinainen pelastustoiminta tapahtuu tehtävätilaksi nimetyssä osuudessa, johon siirrytään hallintatilasta pelastuskokoonpanon määrittelytilan kautta. Pelaajan näkymä on tosiaikastrategiapeleissä tyypillinen lintuperspektiivi eli pelaaja katselee pelikenttää ylhäältä päin. Tehtävätilassa pelaajan tavoite on selvittää kulloinkin tarjolla oleva pelastustehtävä joko sammuttamalla kaikki tulipalot tai pelastaa kaikki tehtävään liittyvät uhrit kuljettamalla heidät sairaalaan. Sammutustyöt voidaan hoitaa joko vaahto-

sammuttimilla tai kytkemällä kuvan 3.3 mukaisesti hyökkäysjohto (attack hose) paloautoon käyttäen auton omaa säiliötä veden lähteenä. Auton lisäksi joissain kentissä on mahdollista käyttää kentiltä löytyviä vesiposteja, joiden kapasiteetti ei ole rajattu toisin kuin paloautoissa.



Kuva 3.2 Konseptikuva paloasemasta (Management mode concept document 2012).



Kuva 3.3 *Letkuselvityksen määrittelykaavio (Hose setup diagram 2012).*

Liikenneonnettomuuksissa mahdollisen sammutustyön lisäksi pelaajan täytyy saada potilaat turvallisesti pois autosta käyttäen pelastuskalustoon kuuluvia erikoisleikkureita. Turvallisen siirron lisäksi pelaajan täytyy stabiloida pelastettava potilas ennen kuin hänet voidaan turvallisesti siirtää sairaalaan käyttäen ambulanssia. Suuremmissa onnettomuuksissa pelaaja voi halutessaan tilata tukiyksiköitä muista simuloiduista pelastusyksiköistä, jos oma tehtävään valittu kapasiteetti ei ole riittävä.

3.1.2 Freeplay-pelimuoto

Freeplay-pelimuoto on julkaisijan pyynnöstä projektiin lisätty pelimuoto, jossa pelataan vain yhdellä kartalla kerrallaan. Freeplay-pelimuodon säännöt on esitetty seuraavassa:

1. Pelaaja kerää pisteitä suorittamalla tehtäviä.
2. Pisteillä pääsee piste-ennätyslistalle.
3. Tehtävissä on aikarajat, joiden umpeutuessa tehtävä merkitään epäonnistuneeksi.
4. Epäonnistuneita tehtäviä voi olla korkeintaan kolme; neljännen kohdalla peli päättyy.
5. Epäonnistuneita tehtäviä voi edelleen suorittaa, mutta niistä ei saa enää pisteitä.
6. Jos pelaaja suorittaa epäonnistuneen tehtävän, tehtävä poistuu listalta, jolloin sitä ei enää huomioida lopetussäännössä (sääntö 4).

3.1.3 Muut tilat

Päätilojen lisäksi Rescue Squad -peliin määriteltiin lukuisia tukitiloja esimerkiksi äänten ja grafiikka-asetusten säätämiseksi. Päätilojen välillä ladataan muun muassa suuria määriä erilaisia graafisia resursseja, johon menee tehokkaallakin pöytätietokoneella useampia sekunteja, jolloin ruutua ei voida päivittää. Pelaaja tulkitsisi tämän pelin kaatumisena tai niin sanottuna jäätymisenä, mikä ei ole toivottavaa. Tähän tarkoitukseen raskaammat operaatiot peitetään päätilojen välissä näytettävillä lataustiloilla. Kyseisissä tiloissa on vain muutamia liikkuvia elementtejä, joilla indikoidaan taustalla tapahtuvaa resurssien latausta.

3.2 Keskeisimmät ominaisuusvaatimukset

Tässä kohdassa kuvataan keskeisimmät pelille määritellyt ominaisuudet ja niiden vaatimukset. Alakohdassa 3.2.1 määritellään pelin muokkaustuen vaatimukset. Muokkaustuen saaminen peliin oli julkaisijalle tärkeää, koska kilpailevissa tuotteissa on ollut saman kaltaisia ominaisuuksia. Alakohdassa 3.2.2 kuvataan pelihahmojen reitinhaun vaatimuksia. Reitinhaku on yksi tosiaikaisten strategiapelien keskeisimpiä ominaisuuksia. Pelaajan pitää kokea hallitsevansa komentamiaan yksiköitä lähes täydellisesti aukottoman pelikokemuksen takaamiseksi. Näin ollen huonosti toteutettu reitinhaku saattaa pilata muuten hyvin toteutetun pelin.

Alakohta 3.2.3 kuvaa pelikenttien vaatimukset. Pelikenttien vaatimukset voivat rajoittaa käytettävissä olevia teknisiä ratkaisuja merkittävästi, joten pelikenttien määrittely on pelin toteutuksen kannalta yksi tärkeimmistä ominaisuusmäärittelyistä. Alakohdassa 3.2.4 määritellään pelitilan tallentamiseen liittyvät vaatimukset. Pelitilan tallennus on pelaajan saaman pelikokemuksen kannalta erittäin tärkeää, joten tämän ominaisuuden toteuttaminen hyvin on pelin kannalta erityisen tärkeää.

3.2.1 Muokkaustuki

Tärkeänä osana projektia nähtiin mahdollistaa käyttäjien itse tekemien paloautojen ja henkilöstön tuonti pelattavaksi. Kyseinen ominaisuus on ollut erittäin suosittu Rescue Squadin kilpailijan Emergency-sarjan peleissä. Erityisesti Emergency 4:n muokkaustoiminnallisuutta on julkaisijalta saatujen tietojen mukaan kiitelty laajasti.

Tavoitteena oli siis päästä muokattavuudessa vähintään edellä mainitun Emergency 4:n tasolle, jossa käyttäjät pystyivät muokkaamaan autoja sekä tekemään uusia pelikarttoja ja tuoda näihin muun muassa itsetekemiään taloja, liikennemerkkejä, aitoja ja muuta grafiikkaa. Tällainen muokattavuus mahdollistaisi käyttäjille mallintaa oman kotiseutunsa paloautot ja niissä käytetyt raidoitukset sekä huomiovalot. Graafisen ulkonäön lisäksi kulkuneuvojen sireenit tulisi olla vaihdettavissa sopivammaksi kulloiseenkin muokkauspakettiin (rescue-mod).

Pääpelin pitää pystyä lataamaan käyttäjän tekemät paketit erillisen muokkausvalikon kautta, joka listaa kaikki tiettyyn asennushakemistoon lisätyt paketit. Nämä paketit puolestaan luodaan erillisellä Fragment Productionin tuottamalla muokkausohjelmistolla. Ohjelmisto lataa käyttäjän valitsemat graafiset ja audioresurssit ja pyytää pelaajaa antamaan esimerkiksi auton tapauksessa toivotut ominaisuudet, kuten ajoneuvon nopeuden ja kiihtyvyyden. Samalla valitaan käytetty sireeni, joko sarjasta valmiita ääniä tai pelaajan tuottamista sireeneistä. Lopuksi pelaaja tallentaa uuden tehdyn paketin Rescue Squad -pelin ladattavaksi.

3.2.2 Reitinhaku

Kuten kaikissa tosiaikaisissa strategiapeleissä, myös Rescue Squad -pelissä pelihahmojen reitinhaku on tärkeässä osassa. Hahmojen on osattava löytää reitti tämän hetkisestä sijainnistaan siihen pisteeseen, johon pelaaja on hahmot komentanut liikkumaan. Tämä ei kuitenkaan vielä riitä. Reitin on oltava lyhin mahdollinen, jolloin hahmo pystyy liikkumaan pelikentän läpi luonnollisen näköisesti. Hahmojen on lisäksi osattava väistää muita hahmoja ja pelikentällä sijaitsevia esineitä, jotta pelaajan illuusio toimivasta pelimaailmasta säilyy.

Rescue Squad -pelissä reitinhaku on tärkeässä roolissa kahdessa pelitilassa, hallintatilassa ja tehtävätilassa. Hallintatilassa pelaaja sijoittelee asemalle kalusteita ja pelaajan ohjaamat pelihahmot käyttävät näitä. Hahmojen on osattava kulkea pelaajan kentälle si-

joittamalta esineeltä toiselle. Hahmojen on myös osattava kiertää reitin varrelle jäävät kalusteet.

Tehtävätilan tuomat vaatimukset reitinhaulle ovat hyvin erilaiset kuin hallintatilan, koska tehtävätilan pelikentät ovat hyvin laajat. Tehtävä saattaa sijaita täysin toisella puolella pelikenttää, missä pelaajan hahmot sijaitsevat. Reitinhaun on toimittava mahdollisimman nopeasti, jotta pelaajan illuusio toimivasta pelimaailmasta säilyy. Hahmoja on myös pystyttävä komentamaan ryhmissä, jotta pelaajan ei tarvitse valita jokaista hahmoa yksitellen ja komennettava näitä liikkumaan tehtävälle. Tehtävätilassa, toisin kuin hallintatilassa, ei ole pelin aikana lisättäviä esineitä. Tehtävätilan pelikentillä on kuitenkin tekoälyn ohjaamaa siviililiikennettä, joita pelaajan hahmojen on pystyttävä väistämään. Reitinhaun teknistä toteutusta kuvataan tarkemmin kohdassa 4.4.

3.2.3 Pelikentät

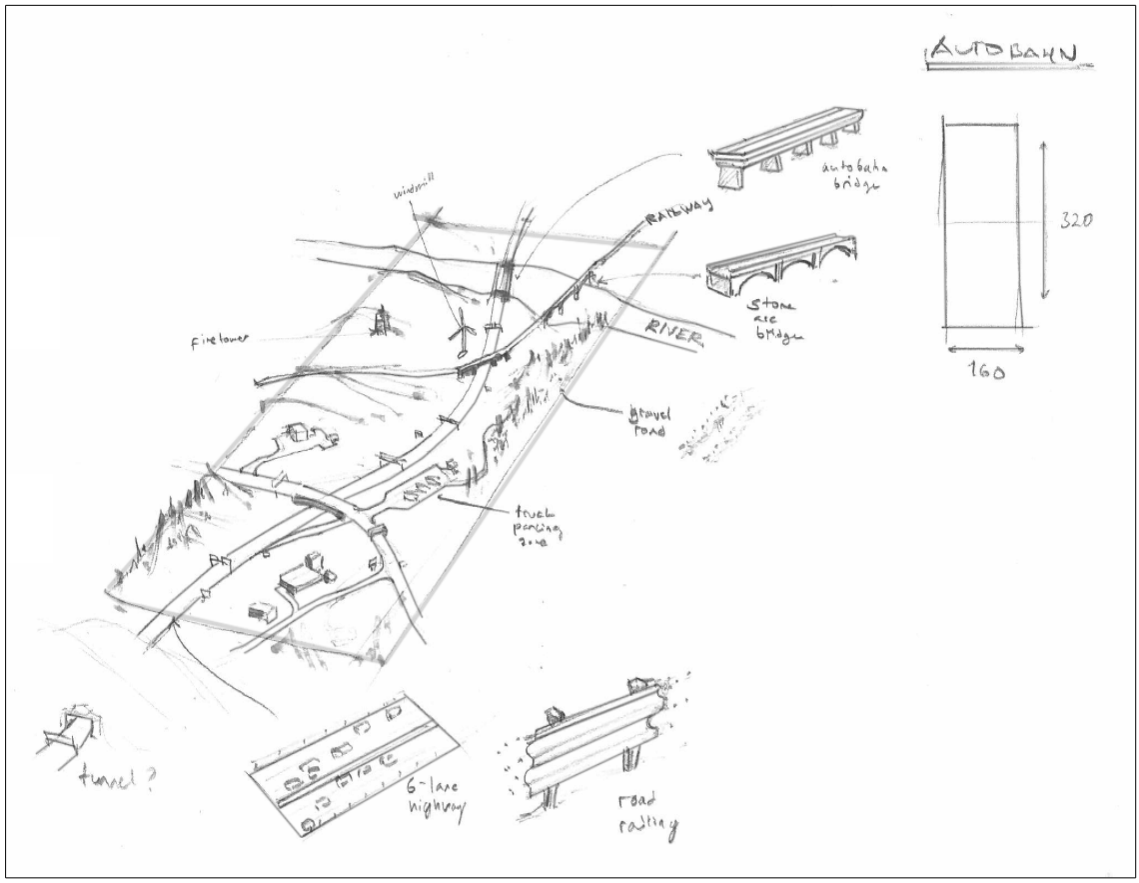
Tehtävätilaa varten Rescue Squad -projektiin täytyi tuottaa tosiaikastrategiapeleille tyypillisiä pelikenttiä, joissa pelaaja suorittaa yksittäisiä pelastustehtäviä. Aluksi vaatimuksena oli koostaa mahdollisimman suuret kentät, mutta tästä luovuttiin, koska liian laajat kentät eivät olisi palvelleet pelillisiä tavoitteita. Lisäksi todettiin, että yli neliökilometrin kokoiset kentät olisivat suuresti rajoittaneet teknisiä toteutusmahdollisuuksia. Lopulta pelikenttien kokovaatimukseksi asetettiin sivuiltaan korkeintaan muutaman sadan metrin suuruiset alueet. Muodoltaan pelikenttien täytyi olla neliöiden lisäksi suorakaiteita eli sivujen pituudet eivät saaneet olla pakotetusti saman suuruiset. Kuvassa 3.4 on Saksa-version moottoritiekenttä, joka määriteltiin olevan 160 x 320 metrin kokoinen.

Koska pelikentät pyrittiin toteuttamaan mahdollisimman realistisiksi, niiden vaatimuksiin lisättiin tarve toteuttaa siviililiikennettä sekä jalankulkijoita. Näitä varten pelikenttiin täytyi pystyä määrittelemään polkuja, joita liikenne ja jalankulkijat seurasivat. Jotta liikenne ei olisi liian ennalta arvattavaa, poluille vaadittiin asetettavaksi solmukohtia eri polkujen välille, jolloin niitä seuraava ajoneuvo esimerkiksi kääntyikin oikealle suoraan ajamisen sijaan.

Aluksi tavoitteena oli saada kenttäsuunnittelijan esimäärittämiä tehtäviin liittyviä tapahtumia, joilla olisi voitu lisätä pelaajan kokemaa dramatiikkaa ja vahvistaa kiireellisyyden tuntemusta. Esimerkiksi tulipalo olisi voinut levitä tiettyyn suuntaan pelaajan toimista riippuen. Dramatiikan lisäksi tämä keino olisi voinut lisätä tehtävien vaikeustasoa sekä uudelleenpelattavuusarvoa. Vaatimus oli kuitenkin suorassa ristiriidassa tehtävien lukumäärävaatimuksen kanssa, joten määrittelyvaiheessa nämä dynaamiset tapahtumat päätettiin rajusti yksinkertaistaa tehtävän alussa ja lopussa esitettäviin tekstimuotoisiin tapahtumakuvauksiin. Pelikenttien toteutus kuvataan alakohdassa 4.3.2.

3.2.4 Pelitilan tallennus

Laajemmissa peleissä hyvin oleellinen ominaisuus on pelitilan tallentaminen, jotta pelaajan ei tarvitse aina palata alkuun, kun peli käynnistetään uudelleen. Vanhoissa peleissä tämä on yleisesti toteutettu pelin sisäisillä koodijärjestelmillä, jolla voidaan palata koodin osoittaman kentän tai tehtävän alkuun. Rescue Squad -projektissa kuitenkin vaatimus oli, että tallennus kirjoitetaan levylle ja pelaaja voi myöhemmillä pelikerroilla ladata valitsemansa tallennuksen ja jatkaa siitä kohdasta, mihin tallennus oli tehty.



Kuva 3.4 *Saksa-version moottoritiekentän konseptikuvaus (Highway-konseptikuva 2012).*

Alun perin tarkoitus oli, että jokaisessa tilassa voitaisiin tallentaa. Näin varmistuttaisiin siitä, että pelaajan etenemistä ei menisi hukkaan. Toteutettaessa tämä kuitenkin osoittautui aikataulullisesti mahdottomasti, sillä tehtävätilan tallentaminen olisi ollut projektin suuruuteen nähden liian aikaa vievä ohjelmointitehtävä. Tästä syystä siitä päätettiin luopua. Huomattavaa on, että tästä syystä myöskään freeplay-pelimuodossa ei voida tallentaa, mikä toisaalta pelillisesti onkin hyvä asia siinä käytetyn pisteytysjärjestelmän takia. Pelitilan tallennus on kuvattu tarkemmin toteutusteknisestä näkökulmasta alakohdassa 4.3.3.

3.3 Julkaistavat versiot

Rescue Squad -projektin lopputuloksena syntyvästä pelistä oli toteutettava useita versioita. Näistä tärkeimmät kuvataan tässä kohdassa. Alakohta 3.3.1 kuvaa pelistä julkaistavien kieliversioiden vaatimuksia. Alakohdassa 3.3.2 kuvataan erityisesti Yhdysvaltoihin suunnatun Steam-latauspalvelussa julkaistun version vaatimuksia.

3.3.1 Kielikäännökset

Suurempien julkaisujen ohessa ilmeni tarve tuottaa myös muita kielikäännöksiä muun muassa puolaksi ja tsekiksi. Lisäksi eräälle itävaltalaiselle yritykselle suunniteltiin yksi ylimääräinen tehtävä, joka oli eksklusiivisesti vain tämän yrityksen kautta myytävissä versioissa.

Tekninen tuki kielikäännöksille oli jo toteutettu aiemmissa tuotetuissa versioissa, joten uusia ohjelmoinnillisia tehtäviä myöhemmät kielikäännösversiot eivät pieniä virhekorjaustehtäviä lukuun ottamatta vaatineet. Muita lokalisaatiossa tapahtuvia ongelmia täytyi kuitenkin ratkoa esimerkiksi puolan ja tsekin kielessä olleiden uusien merkkien takia. Osa näistä merkeistä puuttui aiemmin käytössä olleista kirjaisinatlaksista, joten uudet merkit ja niiden koodit täytyi lisätä käytettävissä olevaan merkistöön.

3.3.2 Yhdysvallat ja Steam-latauspalvelu

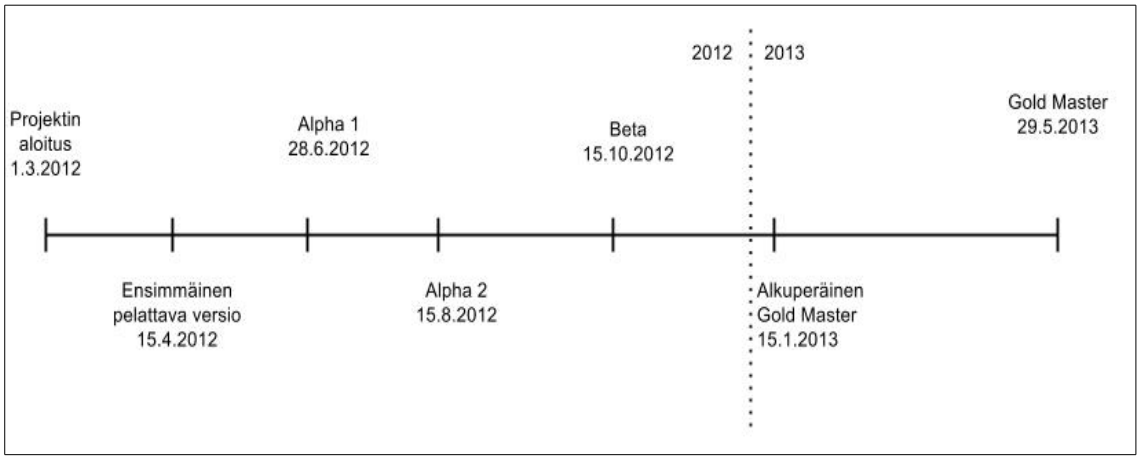
Viimeisin julkaistu versio Rescue Squad -projektista oli suunnattu Yhdysvaltojen markkinoille, mutta toisin kuin aiemmat julkaisut, Yhdysvaltoihin Rescue Squad suunnattiin ainoastaan digitaaliseen jakeluun Steam-latauspalvelun kautta. Yhdysvaltoihin ei siis ollut tarvetta tuottaa erillistä kivijalkamyymälöissä myytävää pakettia. Vaikka Yhdysvallat oli ensisijainen kohdemarkkina, Steamin kautta Rescue Squad -peli tuli saataville maailmanlaajuisesti muillekin markkinoille.

Steam-latauspalvelu toi mukanaan myös uusia teknisiä vaatimuksia projektiin, koska sisäisesti Steam toimii olennaisesti eri lailla kuin aiemmat käytetyt digitaaliset latauspalvelut. Sen sijaan, että palvelu yksinkertaisesti jakaisi samaa DVD:lle painettua pakettia, Steam asentaa pelin käyttäjän puolesta Steamiin valmiiksi määriteltyyn asennushakemistoon. Tämän vuoksi Yhdysvaltoihin tehtyä versiota varten vaatimuksiin kuului myös Steam rajapinnan integrointi Rescue Squad -projektiin.

3.4 Projektin jakautuminen välietappeihin

Julkaisijan vaatimuksesta projekti täytyi jakaa sopimuksentekovaiheessa määriteltyyn viiteen välietappiin. Kohdassa 2.4 esiteltyjen alpha ja beta-välietappien lisäksi Rescue Squad -projektiin lisättiin julkaisijan toiveesta ennen alpha-välietappia ensimmäinen pe-

lattava versio -niminen välietappi. Kuvassa 3.5 on projektin aikataulutus esitettynä aikajanana.



Kuva 3.5 *Rescue Squad -projektin välietapit esitettynä samalla aikajanalla.*

Alakohdassa 3.4.1 kuvataan pelin ensimmäinen pelattava versio ja alakohta 3.4.2 kuvaa pelin alpha-versiota, joka jaettiin julkaisijan toiveesta kahteen välietappiin. Alakohdassa 3.4.3 kuvataan pelin beta-versiota, jossa oli vaatimuksena kaikkien peliin sisältyvien ominaisuuksien toteuttaminen. Pelin valmis versio eli niin kutsuttu gold master on kuvattu alakohdassa 3.4.4. Lisäksi alakohdassa 3.4.5 kuvataan lähes jokaiseen peliprojektiin sisältyvä jälkituotantovaihe.

3.4.1 Ensimmäinen pelattava versio

Ensimmäisen pelattavan version tavoitteena oli esitellä pelin ydinominaisuuksia, kuten tulipalojen leviämistä ja niiden sammuttamista. Kyseisessä versiossa olennaista oli toteuttaa myös ensimmäiset versiot tosiaikastrategiapeleille tärkeistä piirteistä eli kamerajärjestelmä sekä hahmojen valitseminen kartalta ja niiden komentaminen siirtymään pelaajan määrittelemään paikkaan.

Määrittelyä varten erilaisia järjestelmiä yksiköiden komentamiseen oli jo aiemmin tutkittu. Tähän versioon käytettäväksi valittiin Unityn oma navigaatiojärjestelmä, sillä se vaikutti tässä vaiheessa sopivan parhaiten tekniseksi toteutuspohjaksi. Navigaatiojärjestelmän testauksen helpottamiseksi peliin määriteltiin myös väliaikaiseksi tarkoitettu ominaisuus, jolla voitiin tuoda pelitilaan uusia yksiköitä tyhjästä eikä pelaajan siis vielä tarvinnut ostaa tai muuten hankkia niitä.

Ensimmäinen pelattava versio keskittyi pelkästään tehtävätilaan, eli hallintatila siis lisättäisiin myöhemmissä välietapeissa. Määrittely oli hieman epätavallinen, sillä yleisesti ensimmäisestä pelattavasta versiosta olisi hyvä saada kuva koko pelistä eikä pelkästään yhdestä, vaikkakin tärkeästä, ominaisuudesta.

3.4.2 Alpha

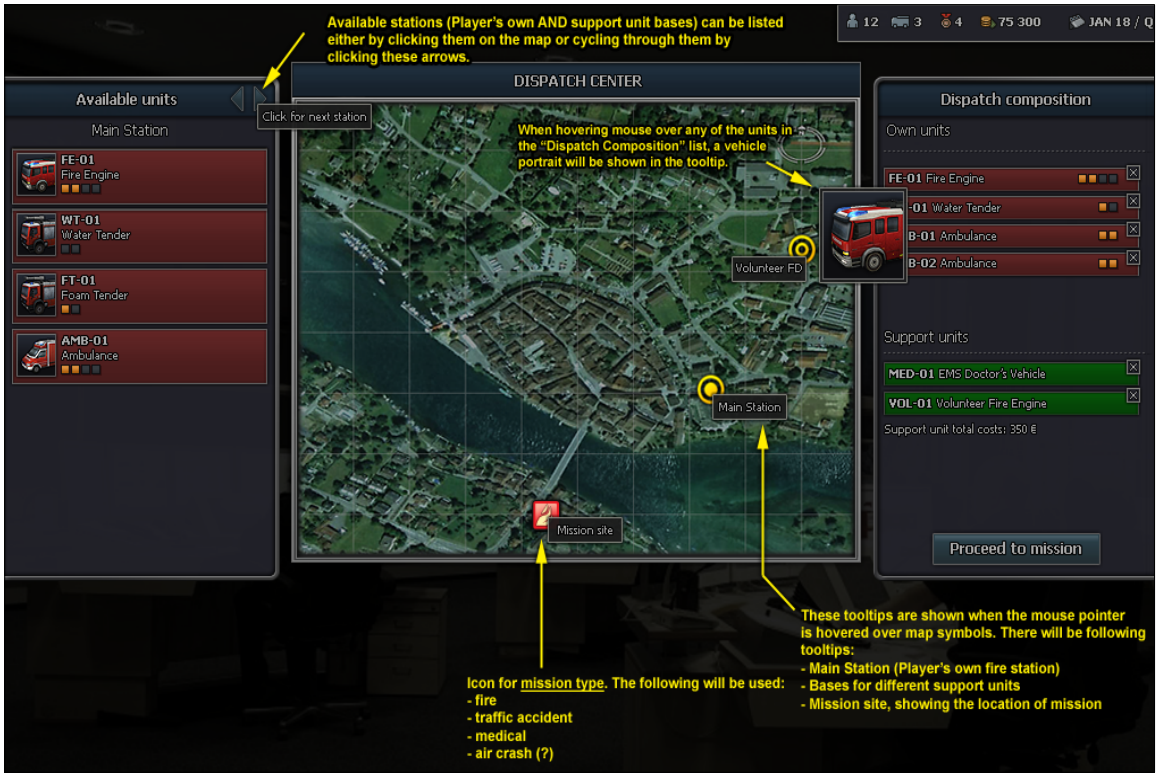
Julkaisijan toiveista alpha-välietappi jaettiin kahteen osaan, jolloin kalenteriaikaa saatiin paremmin tasattua eri välietappien välille. Ensimmäisessä alpha-välietapissa tavoitteena oli edelleen parantaa ensimmäisessä pelattavassa versiossa toimiviksi todettuja ominaisuuksia tehtävätilan puolella, mutta myös tuoda mukaan ensimmäinen versio hallintatilasta. Hallintatilassa tulisi pystyä ostamaan ajoneuvoja sekä palkkaamaan henkilöstä ja lisäksi hankkimaan huonekaluja paloasemalle sijoitettavaksi. Tämä tarkoittaa samalla reitinhakujärjestelmän toteuttamista siten, että hahmot osaavat löytää tekoälynsä avulla kulloiseenkin tehtävään sopivan huonekalun ja hakea sopivan reitin tehtävään sidotulle huonekalulle. Olennainen vaatimus oli, että hahmot pystyvät kiertämään pelaajan asettamat huonekalut eikä vain kulkea niiden läpi aiheuttaen merkittävän graafisen ongelman. Kyseinen ominaisuus oli tässä vaiheessa varsin haasteellinen määritellä aukottomasti, sillä erilaisia erikoistapauksia oli äärimmäisen vaikea ottaa huomioon ilman laajempaa prototypointia. Eräs tällainen oli muun muassa pelihahmon muuraaminen irti muusta paloasemasta siten, ettei hahmolla ollut mahdollisuutta löytää reittiä haluamalleen huonekalulle ilman, että reitti kulkisi jonkun huonekalun läpi.

Hallintatilan ja tehtävätilan välistä yhteyttä ei oltu alpha 1 -välietapissa vielä määritelty, vaan tehtävätilassa toimittiin aina esimääritetyllä pelastuskokoonpanolla. Kyseinen ominaisuus vaadittiin vasta alpha 2 -välietappiin, jossa tuotiin mukaan hallinta- ja tehtävätilan välillä oleva yhdistävä tila, jossa pelaaja valitsee resursseistaan tehtävälle sopivan henkilöstön pelastusajoneuvoineen. Kuvassa 3.6 on esitelty kyseisen kokoonpanonmääritystilan käyttöliittymämäärittely.

Alpha-välietapin vaatimusten laatimisen yhteydessä tehtiin myös kriittistä analyysiä projektin aikataulusta ja lopuista suunnitelluista vielä toteutumattomista ominaisuuksista, kuten pelastushelikopterin ja tikasautojen käyttö. Nämä ominaisuudet päätettiin pudottaa tässä vaiheessa Rescue Squad -projektin ulkopuolelle mahdollisissa jatko-osissa toteutettaviksi. Tosin huomionarvoista on, että varsinkin tikasauton poisjääntі osoittautui pelin julkaisun jälkeen todella kyseenalaiseksi valinnaksi sidosryhmien palautteiden perusteella.

3.4.3 Beta

Beta-välietapin tarkoituksena oli saavuttaa versio, jossa pelin kaikki ominaisuudet ovat toteutettuna ja pelattavissa, mutta eivät kuitenkaan vielä ole viimeistellyssä muodossa. Tämä tarkoittaa, että pelissä voi olla vielä ominaisuuksiin liittyviä virheitä ja epäjohdonmukaisuuksia. Yleisesti pelissä pitäisi olla tässä vaiheessa mukana jo kaikki valmiiseen peliin suunniteltu sisältö, mutta Rescue Squad -projektin tiukan aikataulun vuoksi tästä vaatimuksesta jouduttiin joustamaan. Esimerkiksi kaikkia kenttiä ja niillä suoritettavia tehtäviä ei vielä tässä vaiheessa vaadittu.



Kuva 3.6 *Pelastuskokoonpanon määrittely -näkymän ominaisuusmäärittely (Alpha-vaiheen välietappimäärittelydokumentti 2012).*

Beta-vaiheessa projektin vaatimukset ja aikataulu muuttuivat suhteellisen dramaattisesti, sillä julkaisijan puolelta tuli pyyntö, että peliin lisättäisi vielä yksi kokonainen pelimuoto, jollaista ei siis ollut alkuperäisessä suunnitelmassa. Kyseisen pelimuodon toteuttamisen arvioitiin vaativan 3-4 kuukautta, mikä antoi osaltaan myös hyvin aikaa viimeistellä beta-välietappimääritelmästä pois jääneet tehtävät.

3.4.4 Gold Master

Lopulliseksi julkaisuajankohdaksi lisätyn uuden freeplay-pelimuodon jälkeen määriteltiin toukokuu 2013 alkuperäisen tammikuun sijaan. Aikataulua laajennettiin siis noin neljällä kuukaudella. Gold Master -välietapissa vaatimuksena oli luonnollisesti julkaistava, sisällöltään täysin valmis ja virheetön jälleenmyyntiin kelpaava pelipaketti. Lopullisen tuotteen vaaditut sisältökokonaisuudet on esitetty seuraavassa:

- 36 tehtävää kahdeksalla erilaisella pelikentällä
- 50 hahmoa, joista
  - 40 palomiestä
  - 10 ensihoitajaa
- 13 pelastusajoneuvoa, joista
  - 9 sammutusajoneuvoa

    - 4 ambulanssia
- viimeistellyt äänet ja musiikit
- viimeistellyt ja tasapainotetut kampanja- ja freeplay-pelimuodot
- kattava muokkaustuki (paloautot, hahmot ja tehtävät).

3.4.5 Jälkituotantovaihe

Pelin ensimmäisen julkaistun, Saksaan kohdennetun version jälkeen jälkituotantovaiheessa aloitettiin omat aliprojektinsa Britanniaan, Ranskaan ja USA:han sovelletun version tuotannoista. Nämä maaversiot eivät siis olleet pelkkiä kielikäännöksiä, vaan suurin osa pelin graafisesta ulkoasusta vaihdettiin kuhunkin maahan sopivaksi. Muutettuja kokonaisuuksia olivat muun muassa:

- paloautot (sireenit, 3D-mallit ja tekstuurit)
- pelastushenkilöstö (radioäänet, uusia hahmoja kasvoineen ja taustatarinoineen)
- pelikentät
- tehtävät
- siirtymätilat taustakuvineen
- paloasema.

Yllä mainittujen lisäksi erinäisiä lokalisaatiomuutoksia piti tehdä muun muassa muuttuneen pelin nimen takia. Esimerkiksi Ranska-versioon pelin nimi vaihdettiin ”Rescue 2013: Everyday heroes”:sta ”Rescue Missions D'Urgence”:ksi.

Vaikka kiistatta suurin työ oli graafikoiden puolella tuottaa kaikista muutettavista grafiikkaresursseista uudet versiot, varsinkin ensimmäisen Iso-Britannia -version kanssa työtä oli myös ohjelmointipuolella. Tuotannon alkuvaiheessa tällaisten erillisten maaversioiden tarvetta ei oltu osattu ottaa millään tavalla huomioon, joten tiettyjä valintoja piti miettiä uudelleen ja toteuttaa modulaarisemmin.

Maaversiosta ensimmäisenä tehtiin Iso-Britanniaan sijoittuva versio, jonka yhteydessä tuotettiin myös muun muassa pohjoismaissa myytäväksi tullut niin kutsuttu ”geneerinen Eurooppa” -versio, jossa oli vaikutteita ympäri Eurooppaa, mutta joka ei varsinaisesti leimaantunut mihinkään yhteen alueeseen. Iso-Britannia -version erityishaaste oli maan Saksaan nähden eri puoleinen liikenne. Saksassa liikennesuunta on Suomen tapaan oikeanpuoleinen, kun taas Iso-Britanniassa kadulla ajetaan vasenta kaistaa. Pelaajan paloautot eivät liikennesäännöistä välitä, mutta pelikenttien elävöittämiseksi luodut siviiliautot ja niiden käyttämät polut oli pakko luoda uudestaan tähän versioon. Lisäksi pohjakarttapaloja ei voinut tässä versiossa uusiokäyttää niihin piirrettyjen suuntaviivojen takia.

Seuraavaksi tuotettu Ranska-versio sen sijaan oli huomattavasti helpompi sekä graafisesti että ohjelmointityöltään valmiiden lokalisointityökalujen ansiosta. Samoin graafi-

sen työn määrä oli merkittävästi pienempi, koska ajokaistat olivat samansuuntaiset kuin Saksassa, mikä mahdollisti laajan pohjakarttapalojen uusiokäytön.

3.5 Järjestelmävaatimukset

Pöytätietokonepeliä tuottaessa on aina tärkeää määritellä, millaisella kokoonpanolla lopullisen tuotteen tulisi minimissään toimia. Lisäksi on tehtävä analyysi siitä, kuinka tehokkaan pöytätietokoneen peli vaatii, jotta pelikokemus olisi kehittäjien tarkoittamalla tasolla. Tästä seuraa tavallisesti kaksi eri kokoonpanoa, joista heikompaa kutsutaan minimiksi eli tasoksi, jolla peli juuri ja juuri lähtee käyntiin ja sitä voi jokseenkin tyydyttävästi pelata. Ruudunpäivitysnopeus on tällöin yleisesti hyvin heikkoa, eli silmälle peli näkyy nykivänä ja liike ei ole sulavaa. Suositeltu järjestelmävaatimus on huomattavasti tehokkaampi kuin minimikokoonpano, jolloin peliä on merkittävästi viihdyttävämpi pelata. Liike on sulavaa, latausajat ovat nopeita ja ruutu päivitetään lähes 60 kertaa sekunnissa (fps, frames per second). Ihmissilmä näkee vain noin 24 kuvaa sekunnissa, mutta koska tietokonegrafiikka on vain sarja peräkkäisiä ruutuja, niistä puuttuu tosimaailmassa esiintyvät ilmiöt, kuten liikkeenhämärtyminen (motion blur). Tämän vuoksi tavoiteltu ruudunpäivitysnopeus on nostettu 60 kuvaan sekunnissa 24 kuvanopeuden sijaan.

Rescue Squad -projektissa päädyttiin asettamaan vain minimijärjestelmävaatimukset, jotka olivat noin 4-5 vuotta vanhojen laitteistojen tasolla. Kyseisissä laitteistoissa kuitenkin vaatimuksena oli olla näytönohjain, joka tuki vuonna 2002 julkaistua Microsoftin DirectX- grafiikkarajapinnan versiota 9.0. Monet muut vuonna 2013 julkaistut pelit tosin vaativat jo DirectX 10.0 -versiota (tai uudempaa). Rescue Squad -projektissa kuitenkin pääasiallinen kohdeyleisö oli saksalaiset simulaatiopelaajat, joiden laitteistot todettiin olevan muutaman vuoden muuta pelaajakuntaa jäljessä. Kyseinen analyysi tehtiin tutkimalla muita vastaavia kilpailevia simulaatiotuotteita, kuten vuonna 2012 myöskin Rondomedian julkaisemaa THW-simulator -peliä. Kyseisen pelin järjestelmävaatimukset olivat lähes identtiset kuin mihin Rescue Squad -pelissä lopulta päädyttiin. Taulukossa 3.1 on esillä lopulliset järjestelmävaatimukset.

Taulukko 3.1 *Rescue Squad -pelin järjestelmävaatimukset (Steam 2013).*

Komponentti	Vaatimus
Käyttöjärjestelmä	Windows XP (SP3) / Vista / 7 / 8
Suoritin	Pentium IV CPU 2.0 GHZ tai vastaava
Keskusmuisti	2 GB RAM
Näytönohjain	GeForce 8000-sarjan kortti (DiretX 9.0 yhteensopiva)
Kovalevytila	1 GB vapaata levytilaa
Äänikortti	DirectX 9.0 yhteensopiva

4 PROJEKTIN TOTEUTUS

Tässä luvussa käydään läpi projektin teknistä toteutusta ja toteutuneisiin ratkaisuihin johtaneita syitä. Rescue Squad -projekti toteutettiin käyttäen kohdassa 2.5 esiteltyä Unity Technologies -yrityksen kehittämää Unity-pelinkehitystyökalua. Tämän vuoksi luvussa 4 käsitellään Unityyn liittyviä teknisiä haasteita ja näiden ratkaisuja.

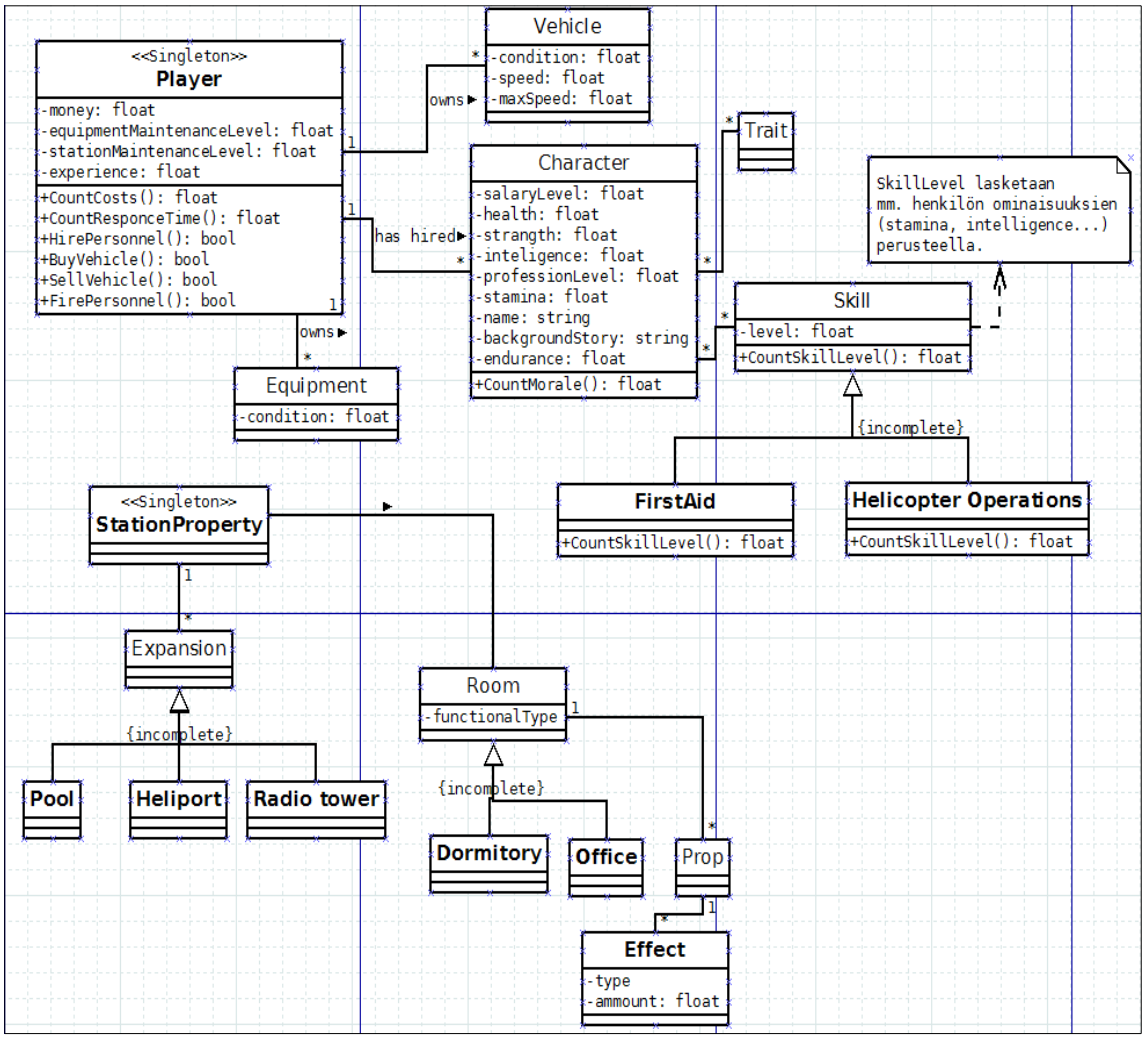
Kohta 4.1 kuvaa Rescue Squad -projektin arkkitehtuuria ja selventää arkkitehtuurisuunnittelussa kohdattuja haasteita. Unity-pelinkehitystyökalua käytettäessä erilaiset laajennukset ovat merkittävässä roolissa, koska näiden ansiosta peliin ei tarvitse toteuttaa itse kaikkia ominaisuuksia. Rescue Squad -projektissa käytettyjä laajennuksia kuvataan tarkemmin kohdassa 4.2.

Kohdassa 4.3 kuvataan peliin määriteltyjen merkittävimpien ominaisuuksien tekninen toteutus. Näiden ominaisuuksien määrittely kuvattiin kohdassa 3.2. Kohta 4.4 kuvaa reitinhaun teknisen toteutuksen. Kohdassa 4.5 kuvataan pelin äänien toteutus teknisestä näkökulmasta. Kohta 4.6 kuvaa projektissa käytetyn ohjelmointiympäristön, ja kohta 4.7 kuvaa projektin lopputuloksena syntyneen pelin markkinoinnissa käytettyjen mainosvideoiden toteuttamisen.

4.1 Käytetty ohjelmistoarkkitehtuuri

Unityä käytettäessä selkeää arkkitehtuurityyliä on vaikea määritellä. Unity on luonteeltaan hyvin geneerinen ympäristö, ja tämä mahdollistaa erityyppisten pelien tekemisen. Haittapuolena on selkeän arkkitehtuurityylin puuttuminen. Unity mahdollistaa esimerkiksi mihin tahansa olemassa olevaan MonoBehaviour:sta periytettyyn olioon viittaamisen, mikä rikkoo kapselointia. Unity myös kannustaa määrittelemään luokkien jäsenmuuttujat julkisiksi. Unityn editori mahdollistaa julkisten jäsenmuuttujien muokkaamisen graafisen käyttöliittymä kautta, mikä helpottaa kenttäsuunnittelijan työtä. Huonona puolena koodista pääsee muokkaamaan näiden muuttujien arvoja, mikä saattaa tuottaa peliin virheitä, varsinkin kun mihin tahansa olioon pystyy viittaamaan pelkän nimen perusteella.

Hyvin suunniteltu arkkitehtuuri selkeyttää ohjelman rakennetta ja täten parantaa ohjelman ylläpidettävyyttä. Vaikka Unity ei helpota hyvän arkkitehtuurin luomista peliin, projektissa pyrittiin kuitenkin tekemään pelin koodista mahdollisimman selkeää ja helposti ylläpidettävää. Seuraavassa kuvataan ohjelmassa käytettyjä suunnittelumalleja. Kuvassa 4.1 on esitetty kampanja-pelimuotoon kuuluvan hallintatilan määrittelyssä käytetty luokkakaavio.



Kuva 4.1 *Suunnitelma hallintatilasta.*

Tämä kohta esittelee tärkeimmät suunnittelumallit, joita käytettiin Rescue Squadin toteutukseen. Alakohta 4.1.1 kuvaa ainokaisen eli singleton-suunnittelumallin ja antaa esimerkin, kuinka tätä suunnittelumallia käytettiin Rescue Squad -projektissa. Alakohdassa 4.1.2 esitellään prorotyyppi-suunnittelumalli ja kuvataan sen käyttöä Unityssä. Alakohta 4.1.3 taas kuvaa rekursiokooste-suunnittelumallin ja esittää, miten Unity soveltaa tätä mallia.

4.1.1 Ainokainen

Ainokainen-suunnittelumallin (singleton) tarkoitus on varmistaa, että luokasta voidaan luoda vain yksi olio ja tarjota globaali tapa käsitellä kyseistä oliota (Gamma et al. 1994, p. 127). Rescue Squad -pelissä ainokaista käytetään erilaisten manager-luokkien toteutukseen. Esimerkiksi SoundManager-luokka huolehtii pelin kaikkien äänten toistamisesta.

Projektiin toteutettiin abstrakti MonoSingletoniksi nimetty luokka, josta kaikki ainokainen-suunnittelumallin toteuttavat luokat periytettiin. MonoSingleton periytettiin MonoBehaviour-luokasta, jotta se voidaan liittää GameObjectiin. Monessa tapauksessa tämä oli tarpeellista, jotta Unityn ominaisuuksia, kuten äänirajapinta, saataisiin käyttöön. MonoSingleton on esitetty koodilistauksessa 4.1.

Ainokaista pidetään yleisesti myös antisuunnittelumallina. Mark Radford (2003) perustelee näkemyksensä ainokaisen olemuksesta antisuunnittelumallina useilla syillä. Eräs syy on se, että hyvän suunnittelukäytännön mukaan olion ei pidä tietää ympäristöstä, jossa sitä käytetään, mitään muuta kuin mitä se saa selville tarjoamansa julkisen rajapinnan kautta. Radford (2003) väittää, että olion tarkoitus on tarjota sovellukselle mahdollisimman suppea rajapinta, joka mahdollistaa olion tarjoaman ominaisuuden saatavuuden sovellukselle. Se, kuinka oliota käytetään sovelluksessa, ei sisälly tähän määritelmään, ja luokka ei näin ollen saisi tehdä päätöstä siitä, montako oliota siitä luodaan.

Eräs oliomallin keskeisimpiä ominaispiirteitä on kapselointi. Kukin luokka huolehtii tietystä hyvin rajatusta vastuualueesta. Tähän vastuualueeseen kuuluvat tiedot ja toiminnot ovat vain sen toteuttavan luokan vastuulla. Globaalit muuttujat rikkovat kapselointinin (Radford 2003), koska niiden kautta muutkin oliot pystyvät muokkaamaan tietoa, jonka pitäisi olla vain tietyn olion muokattavissa. Ainokainen-suunnittelumallin toteuttavat luokat ovat käytännössä globaaleja muuttujia, koska ne tarjoavat vain yhden olion ja tarjoavat siihen pääsyn sen tarjoaman rajapinnan ulkopuolelta (Radford 2003).

Ainokainen tuotti ongelmia myös Rescue Squad -projektissa. Eräs ongelma oli, että kun uusi pelikenttä ladattiin, edellisen pelikentän oliot oli tuhottava. Osa olioista kuunteli ainokaisen lähettämiä tapahtumia, ja kun olio tuhottiin, haluttiin tapahtumien kuuntelu myös lopettaa. Joissain tapauksissa ainokainen oli myös tuhottava uutta pelikenttää ladattaessa, ja kun olio lopetti tapahtuman kuuntelun, oli ainokainen jo tuhottu. Näin ollen tässä tilanteessa ainokaisesta luotiin uusi olio siinä vaiheessa, kun se oli tarkoitus tuhota.

4.1.2 Prototyyppi

Prototyyppi-suunnittelumallin (prototype) ajatus on se, että olioita luodaan prototyypistä kopioimalla (Gamma et al. 1994, p. 117). Kloonattavat luokat periytyvät abstraktista kantaluokasta, jolle on määritelty abstrakti Clone-metodi. Kukin konkreettinen kantaluokasta peritty luokka toteuttaa Clone-metodin niin, että se luo kopion oliosta. Prototyypin etuihin lukeutuu se, että asiakassovellus ei tiedä mitään konkreettisista luokista, joita se luo, vaan se käsittelee prototyyppejä. Näin ollen uuden luokan lisääminen järjestelmään ei vaadi muutoksia asiakassovellukseen.

Unityssä kaikista GameObjecteista voidaan luoda prefab. Prefab on uudelleenkäytettävä GameObject, joka tallennetaan projektiin (Unity 2013d). Prefab voidaan vetää projektin tiedostohierarkianäkymästä tapahtumanäkymään, jolloin prefab:sta luodaan ilmentymä. Prefab toteuttaa prototyyppi-suunnittelumallin.

```
1       public abstract class MonoSingleton<T> : MonoBehaviour where T:
        MonoSingleton<T>
2       {
3         private static T _instance;
4         public static T Instance
5         {
6           get
7           {
8             if ( _instance == null )
9             {
10              GameObject obj = new GameObject( typeof(T).Name );
11              _instance = obj.AddComponent<T> ();
12              _instance.Init ();
13            }
14            return _instance;
15          }
16        }
17
18        public bool DontDestroy = true;
19        protected abstract void Init ();
20        protected virtual void Awake()
21        {
22          if ( _instance == null )
23          {
24            _instance = this as T;
25          }
26          else if ( _instance != this )
27          {
28            Destroy ( gameObject );
29          }
30        }
31
32        protected virtual void Start()
33        {
34          if ( DontDestroy )
35          {
36            DontDestroyOnLoad ( gameObject );
37          }
38        }
39      }
```

Koodilistaus 4.1 *Rescue Squadiin toteutettu MonoSingleton.*

4.1.3 Rekursiokooste

Rekursiokooste-suunnittelumallin (Composite) ajatus on järjestää oliot hierarkkisesti siten, että asiakassovellus voi käyttää koosteita täysin samalla tavalla, kuin yksittäisiä olioitakin (Gamma et al. 1994, p. 163). Rekursiokooste perustuu siihen, että sekä koosteluokka että tavalliset luokat periytyvät samasta kantaluokasta, jossa on määritelty ne toiminnot, joita voidaan tehdä niin koosteille kuin yksittäisille olioillekin (Gamma et al. 1994, p. 163).

Unityn sisäinen hierarkiamalli toteuttaa rekursiokoosteen. Mallin käyttö on olennainen osa Unityn toimintaa pelitiloja (scenejä) tehdessä. Pelitilan muodostavien olioiden hierarkiat koostuvat GameObjecteista, joilla voi olla yksi GameObject vanhempanaan ja useampia GameObjecteja lapsinaan. Jos GameObjectilla ei ole vanhempaa tarkoittaa se, että se on hierarkian juuressa. Unityssä juuressa voi olla useampia GameObjecteja, ja onkin tavallista, että niitä käytetään hieman hakemistojen tapaan jäsentämään hierarkianäkymää. Kun esimerkiksi hierarkiassa yhden GameObjectin siirtää toisen GameObjectin lapseksi, myös kaikki siirrettävän GameObjectin lapset siirtyvät.

4.2 Käytetyt Unity-laajennukset

Unityn laajennustuen ansiosta kaikkia pelin ominaisuuksia ei tarvitse toteuttaa itse. Unity tarjoaa Asset Store -kauppapaikan, jossa kehittäjät voivat kaupata toteuttamiaan laajennuksia. Laajennuksia voi myös hankkia Asset Storen ulkopuolelta.

Kaikki maksulliset laajennukset, joita tarjotaan Unity Asset Storessa, on lisensoitu Unityn omalla lisenssillä, joka mahdollistaa asiakkaalle muun muassa laajennusten käyttämisen kaupallisissa projekteissa. Maksullisten laajennusten lisäksi Unitylle on tarjolla paljon ilmaisia ja avoimen lähdekoodin laajennuksia. Asset Storessa on mahdollista jakaa myös ilmaisia laajennuksia. Unity tarjoaa kolme eri lisenssiä ilmaisen sisällön jakamiseen Asset Storessa (Unity 2013f). Lisäksi mitä tahansa laajennuksia voi tarjota Unityn Asset Storen ulkopuolella millä lisenssiehdoilla tahansa. Seuraavassa on kuvattu tärkeimmän laajennukset, joita projektissa käytettiin. Laajennuksen kuvauksen lisäksi luvuissa kuvataan syyt laajennuksen käyttämiseen.

Alakohta 4.2.1 kuvaa pelin käyttöliittymän toteutuksessa käytetyn NGUI-laajennuksen. NGUI on projektin tärkeimpiä laajennuksia, koska Unity ei tarjoa kunnollisia työkaluja graafisen käyttöliittymän toteuttamiseen. Alakohdassa 4.2.2 kuvataan projektin lokalisointiin käytettyä Localization package -laajennusta. Kuten graafisen käyttöliittymänkin tapauksessa, Unity ei tarjoa myöskään lokalisointiin kunnollisia työkaluja.

Alakohdassa 4.2.3 kuvataan lyhyesti hallintatilan reitinhakuun käytetty A* pathfinding project -laajennus. Tarkemmin pelin reitinhaun toteuttamisesta on kuvattu kohdassa 4.4. Alakohta 4.2.4 kuvaa peliin toteutettujen yksinkertaisten animaatioiden tekoon käytetyt laajennukset. Alakohta 4.2.5 kuvaa pelin sääefektien tekoon käytetyn Unistorm-laajennuksen ja alakohdassa 4.2.6 kuvataan projektin lopputuloksena syntyneen pelin markkinointiin käytettyjen mainosvideoiden tekoon käytettyä Usequencer-laajennusta.

4.2.1 NGUI (Next-Gen UI kit)

Unity sisältää sisäänrakennetun tuen graafisille käyttöliittymille. Kuitenkin jo aivan projektin alkuvaiheessa havaittiin, että Unityn oletustoteutus graafisesta käyttöliittymästä ei ollut projektin tarpeisiin soveltuva. Unityn toteutuksessa jokainen käyttöliittymäelementti aiheuttaa uuden piirtokäskyn. Piirtokäskyjen määrä kannattaa pitää pienenä ja kerrallaan lähetettävä datan määrä suurena (kuitenkin tietyissä rajoissa), koska tietokoneen suoritin muokkaa piirrettävät objektit ensin näytönohjaimelle sopivaan muotoon ja lähettää piirrettävän datan sitten näytönohjaimelle väylää pitkin (Puhakka 2008, s. 165). Puhakan (2008, s. 165) mukaan väylä on tyypillisesti pullonkaula nykyaikaisissa tietokoneissa, joten mikäli suoritin lähettää liikaa piirtokäskyjä näytönohjaimelle, voidaan päätyä tilanteeseen, jossa näytönohjain ei tee mitään ja suoritin yrittää lähettää enemmän dataa näytönohjaimelle kuin väylä pystyy käsittelemään.

NGUI:lla on mahdollista piirtää monimutkainenkin graafinen käyttöliittymä vain yhdellä piirtokäskyllä. Kuva 4.2, jossa on toteutettu keskusteluikkuna, havainnollistaa tätä. Koko käyttöliittymän piirtämisen yhdellä piirtokäskyllä mahdollistaa niin kutsuttu spritejen atlasointi. Sprite on bittikarttakuva, joka sijoitetaan 3D-näkymään (Puhakka 2008, s. 216). Atlas on yksi iso kuva, joka sisältää jokaisen käyttöliittymään piirrettävän spriten. Tehokasta piirtämistä varten spritet on tärkeää järjestää yhteen kokonaiseen atlakseen, jolloin sitä käyttävät monikulmioverkot voidaan piirtää lähettämällä tasan yksi piirtokäsky näytönohjaimelle. Tämä säästö ei ole pöytätietokoneilla merkittävää, mutta on kuitenkin mahdollisia mobiilikäännöksiä silmällä pitäen hyödyllistä tehdä heti ensimmäiseen versioon. Kuva 4.3 esittää pelin kampanjapelimuodon tehtävillä käytettyä atlasta.

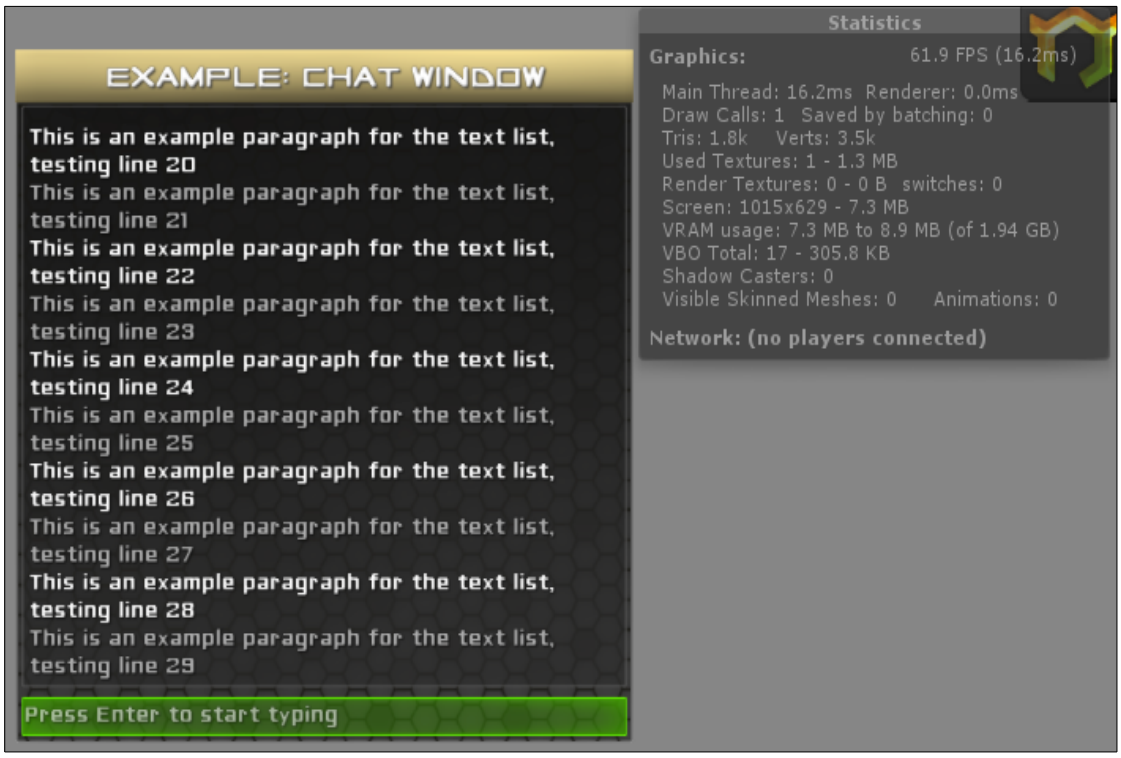
NGUI:ta käytettäessä atlasoinnista on toinenkin hyöty. NGUI:n mukana toimitetaan useita graafisia työkaluja graafisen käyttöliittymän toteuttamiseen. Näiden työkalujen ansiosta muidenkin kuin ohjelmoijien on helppo toteuttaa graafisia käyttöliittymiä. Kun graafisessa käyttöliittymässä käytetyt tekstuurispritet on järjestetty yhteen atlakseen, NGUI:n spriten muokkaustyökalu löytää sen. Tällä työkalulla graafisen käyttöliittymän toteuttajan on helppo lisätä sprite haluamaansa näkymään. Kuva 4.4 esittää NGUI:n spriten muokkaamiseen tarkoitettua työkalua.

Tehokkuuden lisäksi NGUI on myös huomattavasti käytettävämpi kuin Unityn oma käyttöliittymätoteutus. Unityn toteutuksessa käyttöliittymä piirretään aina ohjelmoimalla, minkä vuoksi tarvitaan ohjelmoija toteuttamaan käyttöliittymä. Koodilistauksessa 4.2 on kuvattu, kuinka pelin yhteydessä toteutetussa muokkaustyökalussa on toteutettu tekstikenttä ja yksi painike. Kuva 4.5 esittää koodilistauksen 4.2 toteuttamaa käyttöliittymää. NGUI sisältää monipuoliset työkalut käyttöliittymien toteuttamiseen ja ne integroituvat täysin Unityyn. Näin ollen NGUI:ta käytettäessä graafikko tai kenttäsuunnittelija voi toteuttaa käyttöliittymän ilman ohjelmoijan apua ja ohjelmoijan tehtäväksi jää vain käyttöliittymän toiminnallisuuden toteuttaminen. Tätä ominaisuutta kutsutaan WYSIWYG:ksi (What You See Is What You Get, eli lopputulos on juuri sellainen, miltä se työkalussa näyttää), minkä puute Unityn omassa toteutuksessa on erikoista, kun otetaan huomioon, että Unity on muiden ominaisuuksien osalta panostanut tähän ominaisuuteen. Tämä ongelma on kuitenkin Unity Technologiesin puolelta luvattu korjata tulevissa versioissa (Unity 2013e) tuomalla mukaan täysin uusi käyttöliittymätoteutus.

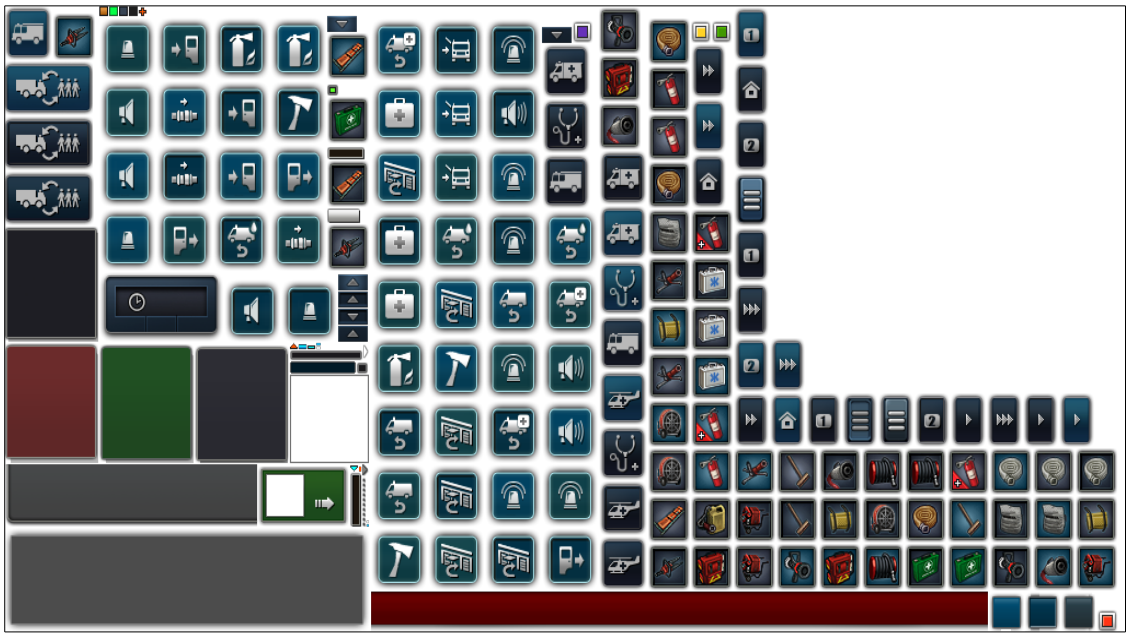
4.2.2 Localization package

Kuten useimmat pelit nykyään, myös Rescue Squad käännettiin useille kielille. Kaikki pelin tekstit kirjoitettiin aluksi englanniksi ja myöhemmin peli käännettiin saksaksi, ranskaksi, puolaksi ja tsekiksi. Koska Unity ei tue lokalisointia suoraan, pitää tuki tälle joko toteuttaa itse tai hankkia laajennus, joka toteuttaa lokalisoinnin. Sopiva laajennus löytyi Unityn Asset Storesta edulliseen hintaan, joten tukea lokalisoinnille ei kannatta-

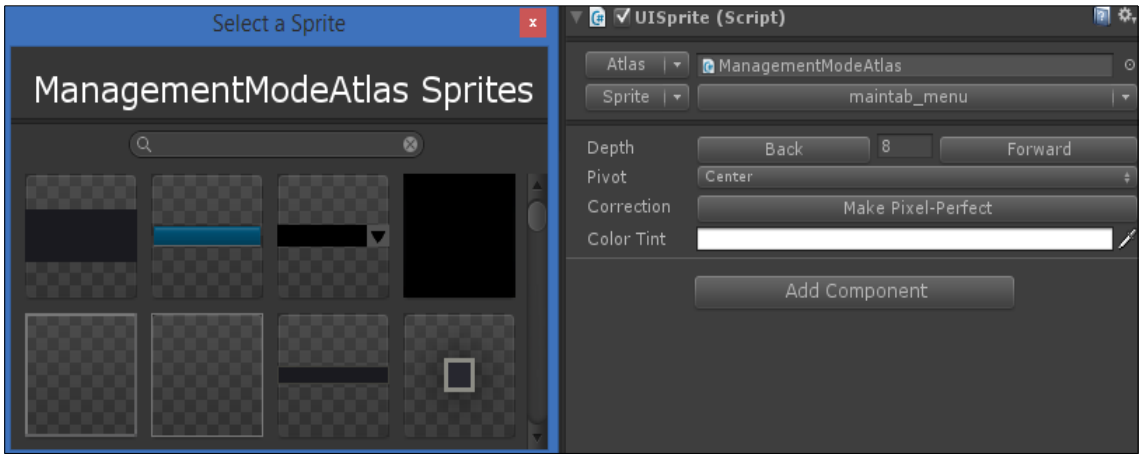
nut toteuttaa itse. Kyseinen laajennus on M2H-nimisen pelistudion kehittämä Localization package.



Kuva 4.2 *NGUI pystyy piirtämään monimutkaisenkin käyttöliittymän vain yhdellä piirtokäskyllä.*



Kuva 4.3 *Esimerkki tekstuuriatlaksesta.*



Kuva 4.4 *Esimerkki NGUI:n tarjoamista graafisista työkaluista.*

Kuva 4.5 *Unityn tarjoamalla käyttöliittymätyökalulla toteutettu tekstikenttä ja painike.*

```
1       void LoadModel()
2       {
3         if ( _isReadyToLoadModel )
4         {
5           GUILayout.BeginHorizontal();
6           {
7             _pathToModel = GUILayout.TextField( _pathToModel,
        GUILayout.MinWidth( 20 ) );
8             if ( GUILayout.Button( "OK" ) &&
        File.Exists( _pathToModel ) )
9             {
10              ModelImporter.Load3d( _pathToModel, false, new
        ModelImporter.LoadCallback( AfterLoadActions ) );
11              _isModelLoaded = true;
12            }
13          }
14
15          GUILayout.EndHorizontal();
16        }
17      }
```

Koodilistaus 4.2 *Esimerkki, kuinka graafinen käyttöliittymä toteutetaan Unityn tarjoamalla työkalulla. Koodi lataa 3D-mallin painiketta painettaessa.*

Localization packagen käyttämän käännökset tallennetaan Google Driven laskentataulukkoon. Kullekin kielelle määritellään omat sarakkeensa. Google Driveen tallennetun taulukon voi jakaa kaikille tiimin jäsenille, joten kuka vain tiimin jäsenistä voi lisätä uusia käännettäviä tekstialkioita peliin tai muokata jo olemassa olevia. Kuvassa 4.6 on esitetty osa projektin käännöstaulukosta.

Käännökset on lisättävä peliin ennen, kuin peli käännetään kohdealustalle. Laajennus lisää Unityyn painikkeen, jota painamalla laajennus lataa käännökset Google Drivesta ja tallentaa ne projektiin XML-muodossa. Käännökset lisätään pelin tekstikenttiin koodilistauksen 4.3 mukaisesti.

			EN	DE	FR
	Comments	Max. length			
GAME_TITLE			Rescue 2013: Everyday Heroes	Rescue 2013: Helden des Alltags	Rescue 2013 : héros du quotidien
MAINMENU_NEW_GAME			Campaign	Kampagne	Campagne
MAINMENU_LOAD_GAME			Load Game	Spiel laden	Charger partie
MAINMENU_OPTIONS			Options	Einstellungen	Paramètres
MAINMENU_EXIT_GAME			Quit	Spiel beenden	Quitter
PAUSEMENU_TITLE			Game paused	Spiel angehalten	Jeu en pause
PAUSEMENU_RESUME			Resume game	Weiterspielen	Reprendre
PAUSEMENU_SETTINGS			Options	Einstellungen	Paramètres
PAUSEMENU_HELP			Help	Hilfe	Aide
PAUSEMENU_ABANDON			Abandon mission	Einsatz abbrechen	Abandonner
PAUSEMENU_RESTART			Restart mission	Einsatz neu starten	Recommencer mission
PAUSEMENU_QUIT			Quit to Windows	Zu Windows	Quitter le jeu
PAUSEMENU_QUIT_OS			Quit to OS	Zum OS	Quitter le jeu
PAUSEMENU_SAVE			Save Game	Spiel speichern	Sauvegarder
PAUSEMENU_MISSION_BRIEFING			Mission briefing	Einsatzbriefing	Briefing
PAUSEMENU_BACK_TO_MAIN_MENU			Back to Main Menu	Zum Hauptmenü	Revenir au menu
WELCOMEPROMPT_HEADLINE			Welcome to Fire Station!	Willkommen auf der Feuerwache!	Bienvenue dans la caserne !
WELCOMEPROMPT_MESSAGE			Greetings, new commander, and welcome to your fire station! Here you manage your personnel and vehicles and decorate the station. Keep your maintenance and tidiness levels high and your personnel motivated and well trained. That way you will be more efficient during missions. Good Luck!	Herzlich Willkommen auf Ihrer Feuerwache! Hier verwalten Sie Ihr Personal, Ihre Fahrzeuge und statten die Wache aus. Sorgen Sie für ein hohes Wartungs- und Sauberkeitsniveau und dafür, dass Ihr Personal motiviert und gut ausgebildet ist. So ist es während den Einsätzen effizienter. Viel Erfolg!	Bienvenue, vous êtes le nouveau chef de centre de cette caserne ! Vous pouvez gérer ici les effectifs, les véhicules et changer la décoration. Maintenez au plus haut les niveaux de maintenance et de propreté, ainsi que la motivation et l'entraînement de vos troupes, pour assurer une efficacité maximale lors des missions. Bonne chance !
WELCOMEPROMPT_BUTTON			I'm ready!	Ich bin bereit!	Allons-y !

Kuva 4.6 *Ruutukaappaus projektissa käytetystä käännöstaulukosta.*

```
1       void UpdateLabelTranslations()
2       {
3         headline.text = Language.Get( "MM_PROPSHOP_HEADLINE" );
4         tabCommon.text = Language.Get ("MM_PROPSHOP_AREA_COMMON");
5         tabWork.text = Language.Get ("MM_PROPSHOP_AREA_WORK");
6         tabRecreation.text =
        Language.Get("MM_PROPSHOP_AREA_RECREATION");
7         tabTraining.text = Language.Get ("MM_PROPSHOP_AREA_TRAINING");
8         tabFacilities.text =
        Language.Get("MM_PROPSHOP_AREA_FACILITIES");
9         totalCost.text = Language.Get ("MM_PROPSHOP_TOTALPRICE_LABEL");
10        currencyLabel.text = Language.Get ("CURRENCY_SYMBOL");
11        purchaseButton.text = Language.Get("MM_PROPSHOP_BUTTON_LABEL");
12      }
```

Koodilistaus 4.3 *Solun sisältö haetaan koodissa listauksen mukaisesti.*

Localization package -laajennuksesta löytyi myös ongelma. Koska pelin on tarkoitus tarjota muokkaustyökalut, pelin kehittäjät halusivat, että käyttäjä pystyisi lokalisoimaan pelin tekstit omalle kielelleen. Localization package -laajennus tallentaa xml-muodossa olevat käännökset Resources-kansioon, joka muunnetaan binäärimuotoon, kun pelistä luodaan julkaisuversio. Tästä syystä käyttäjä ei pysty muokkaamaan lopullisen pelin tekstejä. Ongelman ratkaisemiseksi laajennusta muokattiin siten, että laajennus lukee käännettävät tekstialkiot ohjelman data-hakemistosta olevasta Languages-hakemistosta. Lisäksi ohjelman käännöstyökaluun lisättiin ominaisuus, joka kopioi käännökset sisältävän hakemiston data-hakemistoon, jotta muokattu käännöstyökalu löytää käännöstiedostot. Nyt käyttäjä pystyy muokkaamaan käännöksiä ja halutessaan kääntämään pelin omalle kielelleen.

4.2.3 A* pathfinding project

Pelin hallintatilaan tarvittiin ruudukkopohjainen reitinhakuratkaisu, jotta kalusteiden sijoittelu toimisi järkevästi. Unityn sisäänrakennettu reitinhakujärjestelmä ei tällaista toiminnallisuutta tarjoa. Lisäksi kunnollinen tuki ajonaikaisesti lisättäville esteille tuli Unityn reitinhakujärjestelmään vasta marraskuussa 2013 julkaistussa versiossa 4.3.

Ratkaisuksi tähän ongelmaan päädyttiin käyttämään Aron Granbergin toteuttamaa reitinhakulaajennosta A* pathfinding project. Tämä laajennus kuvataan tarkemmin alakohdassa 4.4.1.

4.2.4 ITween ja HOTween

Pelissä visuaalisesti tärkeä ominaisuus on animaatiot. Esimerkiksi hahmojen liikkuminen näyttää luonnottomalta, ellei samaan aikaan hahmoa liikutettaessa toisteta kävelyanimaatiota. Animaatiot tuovat myös käyttöliittymiin laadun tuntua, kun näkymät eivät vaihdu vain välähtäen. Animaatioiden käyttöönotto vaatii tavallisesti kuitenkin ohjelmoijalta paljon aikaa ja perinteisten animaatioiden tekeminen paljon työtä animaattorilta. Tätä ongelmaa ratkaisemaan on kehitetty animaatiotyökaluja, joiden avulla kenttäsuunnittelijat ja graafikot voivat tehdä yksinkertaisia animaatioita ilman ohjelmoijan tai varsinaisen animaattorin apua. Projektissa käytettiin kahta tällaista kirjastoa, ITweenia ja HOTweenia.

ITween otettiin käyttöön jo aivan projektin alussa. Sillä on toteutettu muun muassa paloautojen ovien avautumis- ja sulkeutumisanimaatiot sekä päävalikon animaatiot. Projektin kuluessa kuitenkin havaittiin, että ITweenista puuttuu useita tärkeitä ominaisuuksia. Esimerkiksi merkkijonojen muuttumista ei pysty animoimaan. HOTween-laajennus tarjosi tämän ominaisuuden, minkä lisäksi se on toteutettu selkeästi tehokkaammaksi kuin ITween. Tämä on helppo todeta HOTweenin sivuilla jaettavasta testistä, jossa sama animaatio on toteutettu sekä HOTweenia että ITweenia käyttäen (Giardini 2013).

Sekä ITween että HOTween tarjoavat graafisen työkalun animaatioiden tekemiseen. Tämän työkalun avulla itse animaatio on helppo tehdä ja animaation voi käynnistää, kun GameObject, johon animaatio kohdistuu, aktivoidaan. Animaation voi myös käynnistää ajastimella. Unityn tarjoamaan animaatiotyökaluun verrattuna tämä on huomattavasti helppokäyttöisempi. Unityn työkalulla animaatio luodaan erilaisia animaatiokäyriä piirtämällä, kun taas ITweenin ja HOTweenin tapauksissa voidaan helposti määrittää, että olio esimerkiksi siirtyy x-akselin suunnassa 800 pikseliä.

4.2.5 Unistorm

Unistorm-laajennuksella on mahdollista toteuttaa erilaisia säätiloja ja valaistuksia vuorokauden ajasta riippuen. Rescue Squad -pelissä Unistormia käytettiin sade-efektin toteuttamiseen.

Alkuperäisen suunnitelman mukaan laajennusta piti käyttää muihinkin sääefekteihin, kuten lumisateen luontiin. Näihin käyttötarkoituksiin Unistorm osoittautui kuitenkin tähän projektiin tarpeettoman monimutkaiseksi ja tästä syystä se jätettiin pois.

4.2.6 USequencer

Usequencer on laajennus, jonka avulla pelimaailman sisällä pystytään luomaan kamera-ajoja ja tallentamaan ne levylle (uSequencer 2013). Kamera-ajoja käytettiin Rescue Squad -projektissa mainosvideoiden tekemiseen.

Varsinaisen pelin osalta Usequencer-laajennusta ei hyödynnetty, vaikka määrittelyvaiheessa mukana projektissa olivat vielä pelitehtävien alussa esitettävät animaatiot. Nämä animaatiot kuitenkin täytyi jättää pois aikataulullisista syistä, koska niiden ei katsottu tuottavan riittävästi pelillistä ulottuvuutta vaadittuun työmäärään nähden.

4.3 Pelin merkittävimpien ominaisuuksien toteutus

Tässä kohdassa kuvataan pelin merkittävimpien ominaisuuksien toteutus. Näiden ominaisuuksien määrittelyt kuvattiin kohdassa 3.2. Alakohta 4.3.1 kuvaa peliin määritellyn muokkaustyökalun toteuttamisen. Kyseisessä alakohdassa kuvataan erikseen autojen ja hahmojen muokkaus.

Alakohdassa 4.3.2 kuvataan, miten pelikentät toteutettiin. Alakohta 4.3.3 kuvaa, kuinka pelitilan tallennus toteutettiin. Kaikkien näiden ominaisuuksien toteutus kuvataan teknisestä näkökulmasta.

4.3.1 Muokkaustyökalu

Projektin tavoitteisiin kuului, että pelaaja pystyy lisäämään peliin itse tekemiään autoja ja hahmoja. Unity ei kuitenkaan tarjoa työkaluja tämän kaltaisen toiminnallisuuden toteuttamiseen. Tuki pelin muokkaamiselle piti tehdä alusta alkaen itse. Aluksi tavoitteena oli, että käyttäjä pystyy luomaan autojen ja hahmojen lisäksi omia karttoja ja tehtäviä. Aikataulusyistä tämä ominaisuus jouduttiin kuitenkin jättämään pois ja lopullinen muokkaustyökalu mahdollistaa vain autojen ja hahmojen tuomisen peliin.

Autojen muokkaus. Autojen osalta käyttäjä voi tuoda peliin 3D-mallin, tekstuurin ja kuvan, jota käytetään auton esittämiseen käyttöliittymässä. 3D-malli tuodaan järjestelmään collada-muodossa. Collada on avoin Khronos Group Inc:n hallinnoima tiedostomuoto. Collada on XML-pohjainen tiedostomuoto, joka on tarkoitettu 3D-grafiikkaresurssien tallentamiseen ja siirtämiseen sovellusten välillä (Collada 2013). Vaikka collada-tiedosto voi sisältää paljon muutakin tietoa kuin itse 3D-mallin, kuten tekstuurin ja animaatioita, muokkaustyökalu lukee tiedostosta vain 3D-mallin. Tekstuuri ladataan työkalulla erikseen. Ovien animaatiot on toteutettu ITween-laajennuksella.

Pelin 3D-mallit on tehty Autodeskin 3ds Max -mallinnustyökalulla ja mallit on tallennettu FBX-tiedostomuotoon. Unity ei kuitenkaan tue FBX-tiedostojen tuontia peliin ajon aikana. Koska FBX on Autodeskin omistama suljettu tiedostomuoto, sille on vaikea luoda toimivaa tuontityökalua. Valmista laajennusta, joka toteuttaa tämän toiminnallisuuden, ei löytynyt, eikä sellaisen tekemiseen itse ollut aikaa. Tämän vuoksi 3D-mallien tuontiin piti valita jokin avoin tiedostomuoto. Vaihtoehdoiksi valikoituivat OBJ- ja collada-tiedostomuodot, koska näiden tuontiin tarkoitettuja laajennuksia oli valmiiksi saatavilla.

OBJ-tiedostomuoto oli ensimmäinen vaihtoehto, jota projektiin harkittiin. Testien perusteella kuitenkin havaittiin, että 3D-mallin keskipisteet (pivot point) eivät säilyneet tuonnin jälkeen siinä kohdassa, johon mallin tekijä oli sen suunnitellut. Keskipiste on se piste, jonka ympäri objektia käännetään. Tämän vuoksi esimerkiksi renkaat eivät olisi pyörineet oikein OBJ-tiedostomuotoa käytettäessä. Collada-tiedostomuotoa testattaessa havaittiin, että 3D-mallit onnistuttiin tuomaan projektiin sellaisena, kuin mallin luonut graafikko oli sen tarkoittanut. Tästä syystä 3D-mallien tallennusmuodoksi valittiin collada.

3D-mallin tuomiseen peliin on kehitetty laajennuksia, mutta yksikään testatuista laajennuksista ei toiminut täysin halutulla tavalla. Parhaaksi osoittautui Jon Martin -nimisen ohjelmoijan avoimen lähdekoodin lisenssillä julkaisema laajennus, joka tuki niin OBJ- kuin collada-tiedostomuotoja (Jon-martin.com 2011). Tämä laajennus oli kuitenkin hyvin keskeneräinen ja se jouduttiin kirjoittamaan suurelta osin uudestaan. Vain itse mallin tuonti ja tallennus Unityn tukemaan formaattiin jätettiin sellaiseksi, kuin se laajennuksessa alun alkaen oli.

Käyttäjän luotua muokattu paloauto, sen tiedot tallennetaan XML-tiedostoon, josta ne voidaan pelissä lukea. Muokatun auton määrittelevä XML-tiedosto on kuvattu koodilistauksessa 4.4.

Muokattu auto voidaan ottaa käyttöön pelin freeplay-pelitilassa. Muokkaustyökalu tallentaa muokatut autot pelin käyttäjäkohtaiseen data-hakemistoon. Freeplay-pelitilan käynnistyessä peli hakee data-hakemistosta kaikki sinne tallennetut muokkaukset. Mikäli muokattuja ajoneuvoja löytyy, peli lukee kunkin auton määrittelevän XML-tiedoston ja luo auton tämän perusteella.

Hahmojen muokkaus. Hahmojen osalta käyttäjä voi tuoda järjestelmään vain uuden tekstuurin ja käyttöliittymässä näkyvän hahmon kuvan. Lisäksi käyttäjä voi määrittää hahmolle nimen, taustatarinan, ammatin ja kokemustason. Muokattujen hahmojen tapauksessa 3D-mallina käytetään samaa mallia, jota käytetään peliin valmiiksi tehtyjen hahmojenkin tapauksessa.

Hahmon muokkauksen toteuttaminen on yksinkertaisempaa kuin autojen, koska projektiryhmä päätti, että hahmon 3D-mallia ei voi vaihtaa. Syy tähän oli se, että muokattuun hahmoon olisi pitänyt toteuttaa samat animaatiot, jotka alkuperäisessäkin mallissa

on. Animaatiot olisi myös pitänyt nimetä identtisesti alkuperäisten kanssa, jotta ne toimisivat oikein. Toinen vaihtoehto olisi käyttää alkuperäisiä animaatioita, mutta tällöin hahmon 3D-mallin on muodostuttava täysin samanlaisesta oliohierarkiasta kuin alkuperäisen hahmon malli.

```
1       <?xml version="1.0" encoding="utf-8"?>
2       <Rescue2013CustomObject name="FireEngineVar" modName="MyFirstMod"
        time="12/18/2013 3:01:31 PM">
3         <name>Example firetruck</name>
4         <charactersslots>4</charactersslots>
5         <description>A basic firetruck.</description>
6         <unittype>FT</unittype>
7         <vehicletype>ModFireVehicle</vehicletype>
8         <model>ExampleFiretruck.dae</model>
9         <rotation>(0.000, 0.707, -0.707, 0.000)</rotation>
10        <scale>(0.800, 0.800, 0.800)</scale>
11        <material texture="ExampleFiretruck.png" shader="Diffuse"
        normalmap="" />
12        <portrait path="Vehicles/Portraits/portrait.png" />
13        <waterSupply amount="1000">
14        <gameObjectName>AttachPoint</gameObjectName>
15          <parent>FireEngineVar_RearDoor</parent>
16          <position>(0.315, 0.253, 0.531)</position>
17          <rotation>(0.000, 0.707, 0.707, 0.000)</rotation>
18        </waterSupply>
19        <pressure>
20          <gameObjectName>Compressor</gameObjectName>
21          <parent>FireEngineVar_RearDoor</parent>
22          <position>(-0.255, 0.253, 0.520)</position>
23          <rotation>(0.000, 0.707, 0.707, 0.000)</rotation>
24        </pressure>
25        <doors />
26        <tires>
27          <tire type="Turning" object="FireEngineVar_FrontWheel02" />
28          <tire type="Turning" object="FireEngineVar_FrontWheel01" />
29          <tire type="Rotating" object="FireEngineVar_RearWheel02" />
30          <tire type="Rotating" object="FireEngineVar_RearWheel01" />
31        </tires>
32      </Rescue2013CustomObject>
```

Koodilistaus 4.4 *Muokkaustyökalun luoma XML-tiedosto muokatun auton tallentamiseksi.*

Hahmon tekstuurin lataamiseen käytetään Unityn WWW-luokkaa. WWW-luokan avulla voidaan ladata resursseja niin internetistä kuin tiedostoistakin. WWW-luokka tukee http-, https-, file- ja ftp-protokollia. Koodilistauksessa 4.5 on kuvattu, kuinka WWW-luokkaa käytetään tekstuurin lataamiseen.

Muokkaustyökalu luo hahmosta XML-muotoisen määrittelytiedoston samaan tapaan kuin ajoneuvostakin. Koodilistaus 4.6 esittää erään muokatun hahmon määrittelevää XML-tiedostoa.

4.3.2 Pelikentän toteutus

Pelikenttien toteutus jaettiin karkeasti kahteen päälinjaan: kentän pohjaosaan ja sen päälle asetettaviin 3D-grafiikkaresursseihin eli esimerkiksi rakennuksiin ja niihin liittyviin tehtäviin. Pohjaosa määrittää myös maastonmuodot, jotka käyttävät Unityn sisäänrakennettua maastotyökalua (Unity 2014b). Maaston muotojen määrittelemiseksi Unityn toteutus käyttää niin kutsuttua korkeuskarttaa, jonka perusteella 3D-verkko luo-

daan (Povray 2014). Kyseinen tekniikka mahdollistaisi useiden satojen neliökilometrien kokoiset alueet (Povray 2014), mutta muista rajoituksista johtuen Rescue Squad -projektissa suurimmat pelikentät olivat kooltaan 320 x 320 metriä. Olennaisin kentän kokoa rajoittava tekijä oli päätyminen tekniikkaan, jossa korkeuskartan perusteella luodun 3D-verkon päälle asetetaan yksi koko kentän pinnan kattava tekstuuri. Kuvassa 4.7 on Steam-version freeplay-pelimuodossa käytettävä pohjatekstuuri. Tekstuuri on alkuperäiseltä kooltaan 8192 x 8192, mutta kuvan versioon se on pienennetty kokoon 512 x 512. Jotta pohjatekstuurin erottelutarkkuus ei laskisi häiritsevän alas, sen kooksi päätettiin 320 metrin kenttien kohdalla 8192 x 8192 pikseliä 24 bitin värisyvyydellä.

```
1       public static Texture2D LoadTextureFromPath ( string path )
2       {
3         if ( File.Exists ( path ) )
4         {
5           path = FixFilePathForUnityWWW ( path );
6           WWW www = new WWW ( path );
7
8           while ( !www.isDone )
9           {
10            Debug.Log ( "Waiting for www to finish." );
11          }
12          return www.texture;
13        }
14        return null;
15      }
```

Koodilistaus 4.5 *Esimerkki Unityn WWW-luokan käytöstä tekstuurin lataamiseen.*

```
1       <?xml version="1.0" encoding="utf-8"?>
2       <Rescue2013CustomObject modName="MyFirstMod" time="12/28/2013
        8:29:17 PM">
3           <firstname>Test</firstname>
4           <lastname>Fireman</lastname>
5           <nickname>Testing</nickname>
6           <age>30</age>
7           <biography>You can add a biography for your firefighter
        here.</biography>
8           <rank>3</rank>
9           <profession>Firefighter</profession>
10          <firemanTexture path="Characters/Textures/fireman_bw.png" />
11          <firediverTexture
        path="Characters/Textures/fireman_firediver_bw.png" />
12          <portrait
        path="Characters/Portraits/portrait_generic_character.png" />
13      </Rescue2013CustomObject>
```

Koodilistaus 4.6 *Muokkaustyökalun luoma xml-tiedosto muokatun hahmon tallentamiseksi.*

Unity ei kuitenkaan suoraan tue 4096 x 4096 suurempia tekstuureja, joten yksi 8192 x 8192 tekstuuri täytyi pilkkoa neljäksi 4096 x 4096 tekstuuriksi. Koska yksi 8192 x 8192 tekstuuri vastasi 320 x 320 metrin kokoista pelikenttää, saadaan pikselisuhteeksi 25,6 pikseliä/metri, joka siis vastasin hyvin lähelle muissa 3D-grafiikkaresursseissa käytettyä pikselisuhdetta. Näin esimerkiksi paloautojen ja rakennusten tekstuurien ja pohjatekstuurin välille ei tullut helposti silmään pistävää erottelutarkkuuseroa. Itse pohjateks-

tuurin tuottamiseen suunniteltiin monimutkainen työkaluketju, mutta koska tämä vaihe toteutuksesta on pääosin graafista työtä, rajataan se tämän diplomityön ulkopuolelle.

Kuva 4.7 *Esimerkki pohjakarttapalasta.*

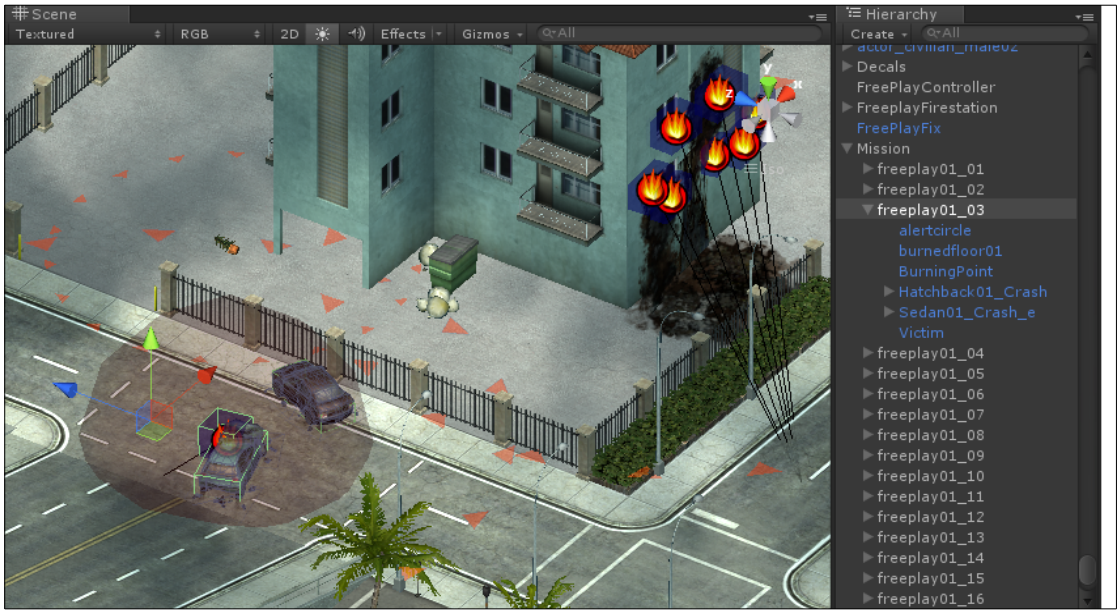
Taulukossa 4.1 on esitelty Rescue Squad -projektissa mahdolliset kenttien koot. Pienimpiä 80 x 80 metrin kenttiä käytettiin vain pienessä määrässä sisätiloihin sijoittuvia pelastustehtäviä, joille suuret ulkotilat eivät olleet tarkoituksenmukaisia. Kuvassa 4.8 esitellään Unityn tapahtumanäkymä, jossa kenttäsuunnittelija viimeistelee pelikentän lopullisen ulkoasun ja siihen liittyvät tehtävät.

4.3.3 Pelitilan tallennus

Pelitilan voi tallentaa vain hallintatilassa. Kaikki data tallennetaan merkkijonoina. Tallentamista helpottamaan projektiin toteutettiin EntityData-luokka. Tämän luokan oliot tallentavat datan Dictionary-tietorakenteeseen. Pelitilan tallentamisen yhteydessä kaikki järjestelmään tallennetut EntityData-oliot haetaan ja niiden Dictionary-tietorakenteiden sisältämät arvot sarjallistetaan ja tallennetaan binäärimuodossa tiedostoon. Sarjallistamisella tarkoitetaan prosessia, jossa data muunnetaan tavuvirraksi, joka voidaan tallentaa esimerkiksi tiedostoon (MSDN 2013). Sarjallistetun datan avulla olio voidaan myöhemmin palauttaa samaan tilaan kuin se oli silloin, kun data sarjallistettiin.

Taulukko 4.1 *Pelikenttien ja niitä vastaavan pohjatekstuurin koot.*

Kentän koko maailman koordinaatistossa (metriä)	Pohjatekstuurin koko (pikseliä)	Tarvittavien tekstuurien lukumäärä
80 x 80	2048 x 2048	1
80 x 160	2048 x 4096	1
160 x 160	4096 x 4096	1
160 x 320	4096 x 8192	2
320 x 320	8192 x 8192	4



Kuva 4.8 *Esimerkkiruutukaappaus Unityn -tapahtumanäkymästä.*

Koodilistaus 4.7 esittelee, kuinka tietoa haetaan EntityDatasta ja tallennetaan EntityDataan siten, että tieto tulee tallennukseen mukaan. Tallennettu pelitila tallennetaan tiedostoon binäärimuodossa. Mikäli Rescue Squad -projekti olisi toteutettu mobiilialustoille, tämä olisi ongelma. Unityn mobiilialustoilla käyttämä versio Monosta ei tue sarjallistamista binäärimuotoon. Mobiilialustoilla sarjallistaminen pitää tehdä XML-muotoon. Dictionary-tietorakenne ei kuitenkaan mahdollista sarjallistamista XML-muotoon, vaan ainoastaan binäärimuotoon. Tulevaisuutta varten onkin hyvä toteuttaa itse IDictionary-rajapinnan toteuttava tietorakenne, joka pystytään sarjallistamaan XML-muotoon.

4.4 Reitinhakujärjestelmä

Tässä kappaleessa tarkastellaan kahta erilaista reitinhakujärjestelmää, jotka päätyivät peliin. Lisäksi tässä kohdassa sivutaan muita vaihtoehtoja, joita ei pelissä kuitenkaan päädytty käyttämään. Alakohta 4.4.1 kuvaa pelin hallintatilassa käytettyä A* pathfinding project -laajennusta ja tämän päälle toteutettua reitinhakua. Alakohdassa 4.4.2 kuvataan Unityn tarjoamaa toteutusta reitinhausta, jota päädyttiin käyttämään pelin tehtävätilassa. Alakohdassa 4.4.3 pohditaan, olisiko jokin muu ratkaisu ollut projektin kannalta mahdollinen.

```
1       private EntityData _data;
2
3       public void AddFunds( float amountToAdd )
4       {
5         float currentFunds = _data.GetAsFloat( "funds" ) + amountToAdd;
6         _data.Add( "funds", currentFunds.ToString() );
7       }
8
9       public void SubstractFunds( float amountToSubstract )
10      {
11        float currentFunds = _data.GetAsFloat( "funds" );
12        currentFunds -= amountToSubstract;
13        _data.Add( "funds", currentFunds.ToString() );
14      }
```

Koodilistaus 4.7 *Pelaajan rahamäärän kasvattaminen ja vähentäminen.*

4.4.1 A* pathfinding project

A* on parhaiten tunnettu algoritmi reitin hakemiseen ja tyypillisesti peliprojekteissa päädytään käyttämään tätä (Pinter 2001). Unityyn on saatavilla useita laajennuksia, jotka toteuttavat A*-reitinhaun. Parhaiten projektin tarpeisiin soveltuvaksi osoittautui Aron Granbergin kehittämä A* pathfinding project (Granberg 2013a). Kyseisestä laajennuksesta on saatavilla kaksi versiota, ilmainen perusversio ja enemmän ominaisuuksia, kuten pelihahmojen keskinäisen väistämisen toteuttavan komponentin sisältävä maksullinen versio (Granberg 2013b).

Yksi tärkeimpiä kokonaisuuksia Rescue Squad -pelissä on hallintatila. Tämä tila simuloi palomiesten arkea asemalla, mistä johtuen pelaajalle haluttiin antaa mahdollisuus kalustaa asema haluamallaan tavalla. Koska palomiehet pystyvät liikkumaan asemalla, reitinhaun on osattava tunnistaa pelaajan asemalle sijoittamat kalusteet ja etsiä reitti väistäen pelaajan lisäämiä kalusteita. A* pathfinding project tukee tätä ominaisuutta hyvin. Järjestelmän perustana on sen alueen kattava ruudukko (Kuva 4.9), jolla hahmot voivat liikkua. Jokainen ruudukon solmukohdista tietää, voiko niiden kautta kulkea vai ei. Kun kaluste asetetaan kartalle, se kysyy reitinhakuruudukolta, mitkä solmukohdat jäävät kalusteen alle. Jos kaikki solmukohdat ovat vapaita, varataan ne ja kaluste jää tälle paikalle. Mikäli yksikin solmukohdista on varattu, ei kalustetta voi sijoittaa kyseiseen kohtaan ja tästä ilmoitetaan pelaajalle.

Kuvassa 4.9 eriväriset viivat kuvaavat niitä reittejä graafin solmukohtien välillä, joiden kautta voi kulkea. Punaiset kuutiot kuvaavat niitä solmukohtia, joilla ei voi liikkua. Siniset, vihreät ja keltaiset neliöt ovat kalusteiden graafista varaamia alueita. Vihreä neliö tarkoittaa aluetta, johon ei voi asettaa toisia kalusteita eikä sen läpi voi kulkea. Sinisen neliön alueelle ei voi asettaa muita kalusteita, mutta hahmot voivat liikkua niiden kautta. Keltaisen neliön kohdalle voi asettaa muita kalusteita ja sen läpi voi kulkea. Aseman seinien leveys on 0,5 metriä, jotta seinä varaa leveyssuunnassa yhden ja vain yhden solmukohdan. Solmukohdat ovat siis 0,5 metrin välein toisistaan. Muutoin haetut reitit näyttäisivät pelaajan näkökulmasta tietyissä tilanteissa epämääräisiltä ja kalusteita ei voisi asettaa aivan seinien viereen.

Kuva 4.9 *Paloasemalla käytetty reitinhakuruudukko.*

4.4.2 Unity-navigaatioverkko ja sitä käyttävät agentit

Unity Technologies lisäsi Unity-pelinkehitystyökaluun sisäänrakennetun reitinhakuominaisuuden versiossa 3.5.0. Kyseinen ratkaisu perustuu navigaatioverkkoon ja sen avulla liikkuviin tekoälyagentteihin. Yksinkertaisimmillaan ohjelmoija antaa agenttikomponentin Destination-kentälle pisteen kolmiulotteisessa avaruudessa, jonka jälkeen Unityn sisäinen toteutus etsii sopivan reitin navigaatioverkkoa käyttäen ja siirtää komponenttiin liitettyä objektia annettua pistettä kohti. Oletustoteutus on kuitenkin tarkoitettu käytettäväksi vain humanoideille pelihahmoille, mikä tarkoittaa, että paloautojen tapauksessa oletustoteutus ei näytä luonnolliselta. Myös hahmojen kohdalla ilmeni pa-

rannettavaa, joten oletustoteutusta piti muokata hahmojenkin kohdalla, eli agenttikomponentti ei päätynyt käyttöön missään tilanteessa sellaisenaan.

Navigaatioverkko itsessään rakennetaan Unityn tapahtumanäkymästä merkkaamalla kentän geometriasta navigoitaviksi (navigation static) ne osat, joilla kenttäsuunnittelija haluaa hahmojen ja autojen voivan liikkua. Merkkaamisen ohessa kenttäsuunnittelija määrittelee geometrialle navigaatiotason, joilla voidaan esimerkiksi estää ajoneuvoja ajamasta tietyille alueille. Estämisen lisäksi navigaatiotasoilla voidaan määrittää niin kutsuttu liikkumisrangaistus eri alueille eli näin voidaan asettaa prioriteetit, joiden mukaan pelaajan yksiköt suosivat tiettyjä reittejä. Olennaisin käyttötapaus kyseiselle ominaisuudelle on saada ajoneuvot suosimaan katuverkostoa rakennusten nurmipihojen sijaan. Joissain tilanteissa pihan kautta oikaiseminen on kuitenkin pelikokemuksen kannalta toivottavaa, joten mainittujen liikkumisrangaistuksien arvojen säätäminen jokaiseen tilanteeseen sopivaksi oli äärimmäisen haastavaa. Kuvassa 4.10 on Steam-versiota varten tehdyn taajamakentän navigaatioverkko esimerkkinä.

Kuvassa 4.10 alue 1 ovat oletustasolla, eli kaikki yksiköt voivat kulkea kyseisillä alueella vain pienellä rangaistuksella. Alue 2:lla puolestaan on merkitty ne alueet, joita ajoneuvot pyrkivät suosimaan, eli niiden liikkumisrangaistus on asetettu minimiin. Alue 3 on esimerkiksi pihoiksi tarkoitettuja alueita, joita ajoneuvot välttävät, eli niiden liikkumisrangaistus on asetettu muita alueita huomattavasti suuremmaksi.

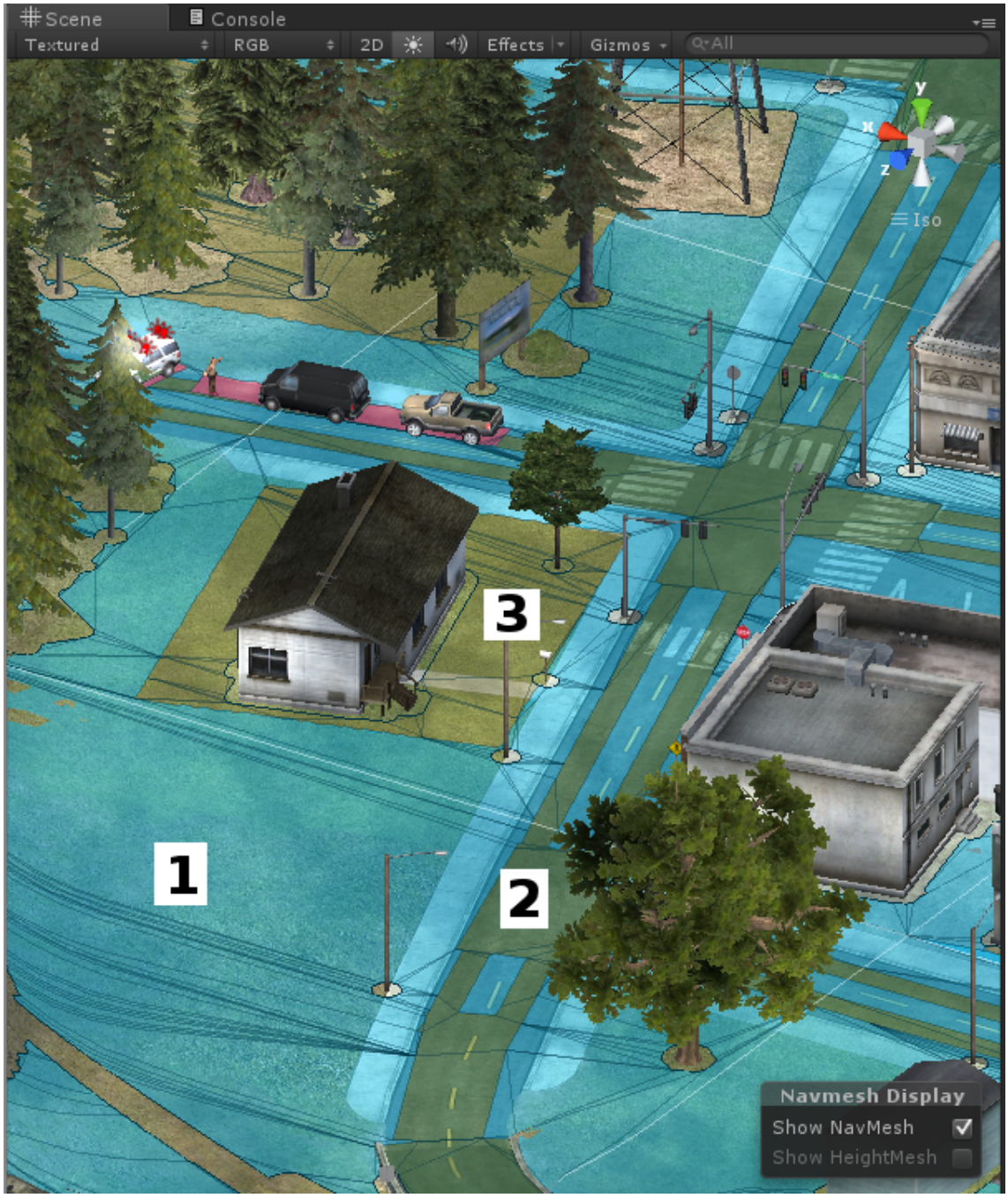
Unityn navigaatioverkkototeutusta käytettiin siis pelin tehtävätilassa, jossa A* pathfinding project osoittautui liian raskaaksi. Hallintatilaan verrattuna käytetyt kentät olivat moninkertaisesti suurempia eli A*:n reitinhakuruudukon solmukohtamäärä olisi kasvanut liian suureksi ja näin reitin hakeminen olisi vaatinut liian paljon laskentatehoa, josta olisi aiheutunut haittaa pelikokemukselle. Navigaatioverkkototeutuksella ei sen sijaan ole tätä ongelmaa, vaan sen avulla suuret tasaiset pinnat voidaan laskea yksittäisinä solmukohtina, mikä laskee merkittävästi vaadittua laskentatehoa reitin hakemiselle.

4.4.3 Muut vaihtoehdot reitinhakujärjestelmän toteuttamiseksi

Edellä mainittujen lisäksi alkuvaiheessa tutkittiin myös muitakin vaihtoehtoja mukaan lukien reitinhaun toteuttamista itse Rescue Squad -projektin aliprojektina. Tämä ajatus oli kuitenkin haudattava, koska tällainen projekti olisi vaatinut erittäin kokeneen peliohjelmoijan, jonka etsimiseen olisi pitänyt käynnistää erillinen rekrytointiprosessi. Tällaiseen ei mitenkään olisi ollut aikaa projektin aikataulun puitteissa.

Unityn kauppapaikassa oli tarjolla useita A* pathfinding projectin ja Unityn oman navigaatioratkaisun kanssa kilpailevia tuotteita. Lupaavimpana oli varsin pitkälle kehitetty Brilliant Game Studios -yrityksen Dynamic Navigation -reitinhakujärjestelmä, joka myöskin on Unity-laajennus. Laajamittaisen testauksen ja prototypoinnin jälkeen kyseinen järjestelmä kuitenkin osoittautui riittämättömäksi Rescue Squad -projektin vaatimuksiin. Ilmaiset reitinhakujärjestelmät puolestaan olivat järjestään joko täysin kesken-

eräisiä tai kauan aikaa sitten hylättyjä, eli niillä ei ollut tukea Unityn uusimpiin versioihin.



Kuva 4.10 *Taajamakentän navigaatioverkko.*

4.5 Musiikki ja ääniefektit

Unity käyttää sisäisesti Fmod-äänijärjestelmää äänen ja musiikin tuottamiseksi. Fmod-äänijärjestelmän lisenssi erikseen hankittuna maksaisi useita tuhansia euroja riippuen peliprojektin laajuudesta (FMOD 2013). Rescue Squad -projektin kohdalla arvioitu kus-

tannus olisi noussut noin kolmeen tuhanteen euroon. Unityn lisenssi kuitenkin automaattisesti sisältää myös Fmod-lisenssin, joten erillistä lisenssiä ei näin tarvita. Fmod sisältää myös äänimuokkaimen, mutta tätä ei Unityn mukana toimiteta.

Unityn äänijärjestelmä tukee niin 2D- kuin 3D-ääniä. 3D-äänet ovat pelimaailmassa kuuluvia ääniä. Rescue Squad -pelissä tällaisia ääniä ovat esimerkiksi paloautojen hälytysäänet ja tulipalon tuottama ääni. 2D-ääniä käytetään valikoissa erilaisten tilasiirtymien tehosteääninä ja osoittamaan, että jotain valikon kohdetta, kuten painiketta, on painettu.

Kapseloinnin kannalta on hankalaa, että Unityllä ei ole yhtä selkeää luokkaa, joka tarjoaa äänirajapinnan. Unityssä ääniä voi toistaa mikä vain MonoBehaviour-luokasta periytetty olio. Mikäli jonkin olion äänentoistossa on virhe, voi tätä virhettä olla vaikea löytää. Rescue Squad -peliin toteutettiin erillinen SoundManager-luokka ratkaisemaan tämä ongelma. Projektiryhmä päätti, että vaikka mikä tahansa pelimaailman olio voi toistaa ääniä suoraan, tätä mahdollisuutta ei käytetä, vaan kaikki äänet toistetaan SoundManagerin kautta. Koodilistaus 4.8 esittelee, kuinka Load-luokka pysäyttää kaikki äänet, kun uusi pelikenttä ladataan. Koodilistaus 4.9 esittää, kuinka musiikki pysäytetään ja käynnistetään uudelleen, kun peli pysäytetään tai kun sitä jatketaan.

```
1       private void StopSounds ()
2       {
3         SoundManager.Instance.StopAmbient ();
4         SoundManager.Instance.StopMusic ();
5       }
```

Koodilistaus 4.8 *Musiikin pysäyttäminen SoundManageria käyttäen.*

```
1       public void TogglePause ()
2       {
3         _paused = !_paused;
4         if (tickerToggled != null)
5           tickerToggled ();
6         if (_paused)
7           SoundManager.Instance.StopMusic();
8         else
9           SoundManager.Instance.PlayMusic("management");
10      }
```

Koodilistaus 4.9 *Pelin pysäyttäminen ja jatkaminen.*

4.6 Ohjelmointiympäristö

Käytännön ohjelmointityö suoritettiin käyttäen Unityn mukana tulevaa MonoDevelopia (versio 2.8). Myöhemmässä vaiheessa käytettiin myös Xamarin Studiota (versio 4.0.x), joka on MonoDevelopista edelleen kehitetty versio, mutta Unity ei ollut päivittänyt omaa Unity-asennuspaketin mukana jaettavaa versiota uudempaan ennen kuin vasta marraskuussa 2013 tulleen 4.3 päivityksen myötä. Näin ollen kehitystyötä hankaloitti kahden eri IDE:n käyttäminen rinnakkain. Tämä oli välttämätöntä, koska Xamarin Studio ei Unity Technologiesin muokkauksien puutteen takia kyennyt jäljittämään virheitä

lähdekoodista. Toisin sanoen lähdekoodi kirjoitettiin pääosin 2.8 versiota uudemmalla MonoDevelopilla ja virheet jäljitettiin ja korjattiin käyttäen 2.8 versiota. Uudemman version käyttäminen lähdekoodin kirjoittamiseen oli lähes pakollista, koska Unityn mukana tulleessa versiossa 2.8 oli vakavia vikoja. Esimerkiksi viittausten etsiminen tiettyyn metodiin kesti todella pitkään riippumatta metodin käytöstä.

Myös Microsoftin Visual Studio 2012 Expressiä kokeiltiin, mutta ilmainen Express-versio oli valitettavasti puutteellinen ammattimaiseen käyttöön. Selvästi hankaloittavin piirre kyseisessä versiossa oli puute laajennuksille, jotka olisivat olleet äärimmäisen tärkeitä tuottavuuden tehostajia. Kaupalliset versiot Visual Studiosta todettiin tämän projektin kohdalla liian hintaviksi. Hinnan lisäksi myöskään Visual Studiolla ei voida oletustilanteessa etsiä virheitä Unity-ohjelmistoista, vaan lisäksi tarvitaan Unity VS -laajennus (UnityVS 2013a), joka on myös kaupallinen tuote ja maksaa noin 200 euroa ohjelmoijaa kohden (UnityVS 2013b).

4.7 Mainosvideot

Pelituotannon aliprojekteina tuotettiin myös monenlaisia markkinointitarkoitukseen suunnattuja tuotoksia, joista tärkeimmät ja laajatöisimmät olivat pelistä kertovat mainosvideot. Videoita tuotettiin kaksi, joista ensimmäinen pyrki tuomaan esiin pelissä esitettyä tunnelmaa palomiehistä arkipäivän sankareina ja jälkimmäinen keskittyi esittelemään itse peliä pelaajan näkökulmasta.

Näihin aliprojekteihin koottiin oma henkilöstönsä vaikka sinänsä suuri osa varsinaisesta työstä tehtiin samaan tapaan käyttäen Unity editoria. Tämä tarkoitti, että mainosvideoissa hyödynnettiin samoja graafisia resursseja kuin mitä pelissäkin käytettiin. Näin voitiin säästää huomattava määrä aikaa. Mainosvideoiden tekoon hyödynnettiin alakohdassa 4.2.6 kuvattua Usequencer-laajennusta.

5 PROJEKTIN ARVIOINTI

Tässä luvussa arvioidaan projektin vaatimusten toteutumista. Kohdassa 5.1 arvioidaan kunkin välietapin onnistumista. Kohdassa 5.2 käydään läpi projektin jälkituotantovaihetta, joka Rescue Squad -projektin tapauksessa oli erittäin merkittävässä roolissa. Kohdassa 5.3 arvioidaan projektin tärkeimpien teknisten ominaisuuksien onnistumista. Kohdassa 5.4 on esitelty ne ominaisuudet, jotka projektiin oli alun alkaen suunniteltu, mutta joista jouduttiin muun muassa tiukan aikataulun vuoksi luopumaan.

5.1 Välietappien toteutuminen

Kuten kohdan 3.4 vaatimuksissa esiteltiin, pääprojekti jaettiin viiteen eri välietappiin, joiden välissä oli ajalliset noin 2-3 kalenterikuukautta. Alakohdassa 5.1.1 arvioidaan ensimmäisen pelattavan version onnistumista ja erityisesti sen merkitystä projektille. Alakohta 5.1.2 arvioi, miten alpha-välietappi toteutui ja alakohta 5.1.3 arvioi beta-välietapin toteutumista. Tärkeimmän välietapin, gold masterin, onnistumista arvioidaan alakohdassa 5.1.4.

5.1.1 Ensimmäinen pelattava versio

Määritelmän mukaiset ominaisuudet toteutettiin aikataulun mukaisesti, eli teknisesti ensimmäinen pelattava versio voidaan katsoa onnistuneeksi. Kuitenkin pelattavassa versiossa paljastui suuria ongelmia erityisesti palosimulaatioon liittyen. Havaittiin, että realistinen tulipalo huoneistopaloissa ei ole pelillisesti mielenkiintoista liiallisen aikakriittisyyden vuoksi. Huoneistoissa tulipalo leviää pelin kannalta liian räjähdysmäisesti. Kuvassa 5.1 pelitilanne on edennyt noin puoli minuuttia ja tilanne on pelaajan kannalta jo täysin toivoton. Toisaalta simulaatiomielessä tilanne on tehdyn taustatyön perusteella varsin todenmukainen, koska usein huoneistopalojen yhteydessä pelastushenkilöstöllä ei ole enää mitään tehtävissä palon leviämisen estämiseksi.

Olennaisin johtopäätös ensimmäisen pelattavan version perusteella oli, että määrittely vaatii vielä suuren määrän tarkennuksia ja iteraatioita. Muun muassa huomattiin, että realistisen palon leviämistä on hyvin vaikea kommunikoida pelaajalle ilman suurta määrää graafista työtä. Graafisen esityksen sijaan tuleviin versioihin määriteltiin, että tulipalopesäkkeisiin liittyvä tieto esitetään käyttöliittymän tasolla.

5.1.2 Alpha

Alpha-välietapissa olennaisin tavoite oli toteuttaa ensimmäinen iteraatio pelin hallintatilasta. Alpha 1 -välietapissa tämä toteutettiin vielä pääosin väliaikaisella käyttöliittymällä, joka myöhemmin korvattiin julkaisuun päätyneellä versiolla. Aikataulun tiukkuuden vuoksi osaa väliaikaisesta toteutuksesta ei ehditty vaihtamaan. Kuvassa 5.2 on tilanne hallintatilasta alpha 1 -vaiheesta, jossa on vielä ensimmäisen iteraation käyttöliittymäelementtejä. Käyttöliittymä itsessään ei kuitenkaan ollut vaiheen suurin prioriteetti, vaan se pyrittiin toteuttamaan mahdollisimman nopeasti, jotta alpha-välivaiheessa toteutettavat pääominaisuudet saataisiin testattua. Vaiheessa tärkeintä oli, että pelaaja pystyi hankkimaan paloasemalle kalusteita, henkilöstöä ja pelastusajoneuvoja, sekä lähettämään pelastuskokoonpanon erillisen määritystilan kautta tehtäville. Kyseinen määrittelytilan tehtävä on siis linkittää hallinta- ja tehtävätilan yhteen. Koska vaiheen päätteeksi pelin runko täytyi olla kokonaisuudessaan pelattavissa, ominaisuusmäärittelyyn lisättiin vaatimus pelin tallentamisesta. Tämä oli tärkeää toteuttaa jo tässä vaiheessa, sillä pelin tehtävien sisällön tuotanto alkoi myös tässä vaiheessa. Tehtäväketjujen kunnollisen testauksen mahdollistamiseksi tallennusominaisuus oli siis kriittisen tärkeä.

Kuva 5.1 *First playable.*

Alpha 2 -välietapin olennaisin lisäys hallintatilaan tehtävien parannusten ja kokoonpanonmääritystilan lisäksi oli esitellä peliin liikenneonnettomuudet, jotka olivat kokonaan uusi tehtävätyyppinsä. Liikenneonnettomuuksien määrittely jakautui kahteen osaan: uhrien pelastaminen autoista ja heidän kuljettaminen sairaalaan stabiloinnin jäl-

keen. Tämä tarkoitti kahden uuden pelastusajoneuvon lisäämistä. Raivausauton ja siihen kuuluvan henkilöstön tehtävä oli saada uhrit irti autoista, jonka jälkeen ambulanssilla paikalle tuodun ensihoitohenkilökunnan tehtävä oli kuljettaa potilaat sairaalaan. Kuvassa 5.3 on esitelty alpha 2 -välietappia varten toteutettu liikenneonnettomuustehtävä.

Kuva 5.2 *Pelin hallintatila, jossa asemalle on hankittu kolme autoa ja kalusteita. Pelin tallentaminen oli jo toteutettu alkeellisella tasolla.*

5.1.3 Beta

Alkuperäiseen beta-välietappimäärittelyyn ei kuulunut enää juurikaan uusia teknisiä ominaisuuksia, vaan ohjelmointipuolella suurin työ keskittyi löydettyjen virheiden korjaamiseen ja olemassa olevien ominaisuuksien paranteluun. Eräs tällainen ominaisuus

oli lisätä tehtävätilaan koko aluetta esittävä pienoiskartta, joka helpotti varsinkin suurissa kentissä navigointia. Pienoiskartta on esillä kuvan 5.4 oikeassa yläreunassa.

Virheiden korjaus osoittautui kuitenkin ennakoitua työläämmäksi tehtäväksi. Projektissa oli jo aiemmin ollut tiettyjä ongelmia, jotka oli aiemmissa välietapissa mahdollista ohittaa, mutta erityisesti Gold Master -välietapien lähestyessä ongelmat oli kriittistä saada korjatuksi jo beta-välietappiin. Suuritöisimpiä näistä ongelmista oli NGUI-laajennuksen ja DirectX-piirtorajapinnasta aiheutuneiden piirtovirheiden korjaaminen. Kyseiset ongelmat on esitetty tarkemmin alakohdassa 5.3.3.

Kuva 5.3 *Alpha 2 -välietapissa toteutettu liikenneonnettomuuden selvittäminen.*

Beta-välietapin toteuttamisen aikaan ilmeni tarve muuttaa projektin aikataulutusta, koska julkaisija halusi tässä vaiheessa lisätä peliin vielä yhden kokonaan uuden freeplay-pelimuodon. Pelimuodossa tehtäviä suoritetaan vain yhdellä pelikentällä käyttäen esimääriteltyä pelautushenkilökuntaa ja -kalustoa. Ominaisuusmäärittely pyrittiin laatimaan siten, että kyseinen uusi pelimuoto vaatisi mahdollisimman vähän uusia teknisiä ominaisuuksia. Syy tähän oli yksinkertaisesti se, että vaatimus esitettiin niin lähellä projektin alkuperäistä Gold Master -välietappia, johon oli aikaa enää noin kaksi kuukautta. Lopulta uusi pelimuoto laajensi aikataulua neljällä kuukaudella eli lopulliseksi Gold Master -välietapin takarajaksi asetettiin 29.5.2013.

5.1.4 Gold Master

Gold Master -välietappi päättää projektin varsinaisen tuotantovaiheen, eli vaatimuksena oli sisällöltään täysin valmis ja virheetön julkaisupaketti. Alakohdassa 3.4.4 esitetyistä vaatimuksista vain muokkaustuki jäi puuttumaan ensimmäisestä julkaisusta versioista.

Vaatimusmäärittelyä muutettiin siten, että muokkaustuki siirrettiin toteutettavaksi jälkituotantovaiheeseen tasapainotus- ja virhekorjauspakettien yhteydessä. Gold Master -välietappia toteutettaessa todettiin, että muokkaustukea tärkeämpää oli julkaista mahdollisimman hiottu ja tasapainotettu ensimmäinen versio, joten työpanos fokusoitiin pienimpienkin virheiden korjaamiseen. Kuvassa 5.5 on esitetty ruutukaappaus eräästä Gold Master -version sammutustehtävästä.

Kuva 5.4 *Ruutukaappaus beta-välietapissa tuotetusta versiosta.*

5.2 Jälkituotantovaihe

Iso-Britannia- ja Ranska-maaversiot eivät vaatineet juurikaan uusia ominaisuuksia lokalisointityötä lukuun ottamatta, joten ne olivat teknisesti hyvin triviaaleja, joskin työläitä, toteuttaa. Kaikki jälkituotantovaiheessa toteutetut versiot julkaistiin aikataulun mukaisesti pitäen sisällään kaikki määritellyt ominaisuudet ja sisältökokonaisuudet. Kuitenkin ensimmäiseen Gold Master -versioon jäi joitakin virheitä, jotka oli korjattava Saksaan julkaistuun versioon. Virheet sinänsä olivat suhteellisen merkityksettömiä, eli minkäänlaisia pelissä etenemistä estäviä virheitä ei löytynyt, eikä myöskään peliä yllättävästi sammuttavia ongelmia ilmennyt. Jälkituotantovaihe päättyi Steam-version julkaisuun

11.12.2013, joka mahdollisia korjauspäivityksiä lukuun ottamatta päättää Rescue Squad -projektin. Kuvassa 5.6 on esillä Rescue Squadin USA -version Steam-latauspalvelun kauppasivu.

Kuva 5.5 *Ruutukaappaus valmiista Gold master -versiosta.*

Pelistä piti tehdä myös Japaniin suunnattu oma maaversionsa, mutta tuotantotaloudellisista syistä tämä versio jäi ainoastaan suunnitteluasteelle. Tähän johti UK-versiotakin vaivannut ongelma vasemmanpuoleisesta liikenteestä, joka yksinkertaisesti olisi tuottanut liiaksi työtä, ja tämän myötä kulut olisivat nousseet sellaisiksi, ettei projekti olisi ollut kannattava kaupallisessa mielessä. Ongelmaa lisäsi vielä japanin kielen länsimaisiin kieliin verrattuna täysin erilainen kirjoitusasu, joka olisi vaatinut muun muassa kaikkien valikkotilojen uudelleen asemoinnin. Kaiken kaikkiaan projektin aikana Resque Squadista julkaistiin yhdeksän erillistä versiota, jotka on listattu taulukossa 5.1 yhteenvetona.

Taulukko 5.1 *Kaupalliseen levitykseen julkaistut versiot.*

Versio	Julkaisunimi	Julkaisupäivä	Huomioita
Saksa, perusversio	Rescue 2013: Helden des Alltags	29.5.2013	-
Saksa, erikoispainos (keräilyversio)	Rescue 2013: Helden des Alltags - Collector's Edition	29.5.2013	Myyntipakettiin lisätty oheistuotteita, kuten paloletkusta tehty avaimenperä.

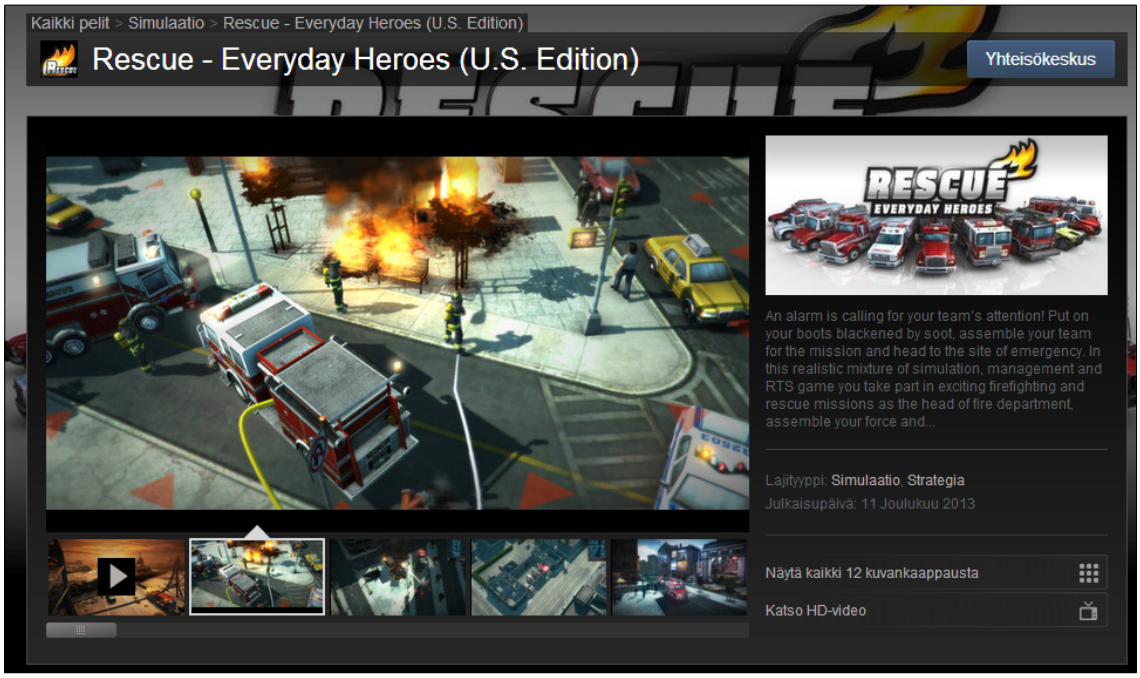
Versio	Julkaisunimi	Julkaisupäivä	Huomioita
Itävalta	Rescue 2013: Helden des Alltags	29.5.2013	Saksan perusversio, johon on lisätty yksi ylimääräinen tehtävä.
Iso-Britannia, perusversio	Rescue 2013: Everyday Heroes	5.7.2013	-
Iso-Britannia, geneerinen versio	Rescue 2013: Everyday Heroes	5.7.2013	Yleiseurooppalainen versio, jossa oli sisältöä sekalaisesti eri maaversioista.
Ranska, perusversio	Rescue Missions d'Urgence	4.9.2013	-
Tsekki	Rescue 2013: Everyday Heroes	29.11.2013	Geneerisen perusversion kielikäännös.
Puola	Rescue 2013: Prawdziwi bohaterowie	29.11.2013	Geneerisen perusversion kielikäännös.
Yhdysvallat	Rescue - Everyday Heroes (U.S. Edition)	11.12.2013	Myytävänä vain Steam-latauspalvelun kautta.

5.3 Tekninen arviointi

Alakohdassa 5.3.1 arvioidaan muokkaustuen onnistumista ja 5.3.2 kuvaa ongelmia Unityn päivittämiseen liittyen. Alakohdassa 5.3.3 puolestaan analysoidaan NGUI:n käyttöä projektissa.

5.3.1 Muokkaustuki

Muokkaustuki Rescue Squad -peliin toteutettiin projektin loppuvaiheessa. Aikataulullisista syistä johtuen muokkaustyökalusta jätettiin kartta- ja tehtävätoiminnallisuus kokonaan pois. Jäljelle jäi autojen ja hahmojen muokkaus. Alkuperäisen suunnitelman mukaan muokkaustuen piti olla valmis samaan aikaan kuin varsinainen pelikin ja muokkaustyökalu oli tarkoitus toimittaa samalla asennusmedialla pelin kanssa. Pelin aikataulu oli kuitenkin niin tiukka, että muokkaustyökalua päästiin toteuttamaan vasta aivan projektin lopussa. Muokkaustyökalua ei ehditty saamaan valmiiksi, kun valmis versio pelistä toimitettiin julkaisijalle. Koska julkaisija oli ehtinyt jo markkinoida peliä muokkaustuella, piti muokkaustyökalu saada pelin ostajien saataville julkaisupäivänä. Ongelma päätettiin ratkaista jakamalla muokkaustyökalun toteutus kolmeen vaiheeseen. Näin ensimmäinen versio muokkaustyökalusta ehdittäisiin saamaan valmiiksi ja käyttäjille internetin kautta ladattavaksi pelin julkaisupäivänä.



Kuva 5.6 *Rescue Squadin Steam-kauppasivu (Steam 2013). Yhdysvalloissa peli julkaistiin nimellä Rescue - Everyday Heroes (U.S Edition).*

Muokkaustyökalun tärkein ominaisuus on autojen muokkaus. Näin ollen muokkaustyökalun ensimmäiseen versioon päätettiin toteuttaa autojen muokkaus. Projektiryhmä piti riskialttiina sallia muokattujen autojen käyttö kampanjatilassa sen monimutkaisuuden takia. Freeplay-pelimuodon lisääminen peliin projektin loppuvaiheella ratkaisi tämän ongelman. Muokattujen autojen käyttö päätettiin sallia ainoastaan freeplay-pelimuodossa.

Muokkaustyökalun toiseen versioon lisättiin hahmojen muokkaus. Alkuperäisen suunnitelman mukaan myös hahmojen 3D-mallin tuominen piti olla mahdollista. Animaatioiden toteuttaminen muokatuille hahmomalleille olisi kuitenkin ollut erittäin vaikea ongelma ratkaistavaksi. Ominaisuuden toteuttamisen vaikeuden lisäksi muokattujen hahmojen tekijöille pelissä toimivien animaatioiden tekeminen olisi ollut erittäin vaikeaa ja näin ollen muokkaustyökalun käyttökynnys olisi noussut liian korkeaksi. Ongelma ratkaistiin siten, että hahmojen 3D-mallin tuomisesta peliin luovuttiin ja vain hahmon tekstuurin muokkaaminen sallittiin. Muokatuille hahmoille on myös mahdollista määrittää nimi, ikä, taustatarina, ammatti ja kokemustaso. Lisäksi työkalu käännettiin saksaksi ja ranskaksi tässä vaiheessa.

Muokkaustyökalun kolmannessa versiossa päätettiin parantaa työkalun käytettävyyttä. Kahdessa ensimmäisessä versiossa esimerkiksi 3D-mallien ja tekstuurien polut piti kirjoittaa manuaalisesti. Kolmanteen versioon toteutettiin graafinen tiedoston valinta -ominaisuus, joka parani työkalun käytettävyyttä merkittävästi. Lisäksi jo muokattujen

hahmojen lataaminen ja edelleen muokkaus toteutettiin tähän versioon muokkaustyökalusta. Kuva 5.7 esittää muokattua autoa freeplay-pelimuodossa.

Kuva 5.7 *Muokattu auto otettu käyttöön freeplay-pelimuodossa.*

Muokkaustuen toteutus osoittautui varsin työlääksi, eikä sitä lopulta pystytty toteuttamaan siinä laajuudessa kuin alun perin oli suunniteltu. Suurin ongelma oli se, että Unity ei tue tämän kaltaisia ominaisuuksia millään tavalla. Esimerkiksi uusien 3D-mallien tuominen peliin on toteutettava itse ja Unityn omia tietorakenteita ei pysty käyttämään näiden tallentamiseen. Jälkikäteen arvioituna muokkaustuen kehitykseen käytetty aika olisi ollut parempi käyttää pelin muiden ominaisuuksien parantamiseen ja tulevaisuudessa tämän kaltaisten ominaisuuksien toteuttaminen on oltava oma projektinsa. Lisäksi Unityn käyttöä tulevissa projekteissa on harkittava tarkkaan, mikäli muokkaustuki on näissä merkittävässä roolissa.

5.3.2 Unityn migraatio versiosta 3.x versioon 4.x

Projektin alussa Unitystä oli käytössä versio 3.5. Tämä oli ensimmäinen Unityn versio, joka tuki reitinhakua. Reitinhaku oli tässä versiossa kuitenkin hyvin alkeellinen. Reitinhaku ei esimerkiksi tukenut suorituksen aikana lisättäviä esteitä. Lisäksi Unityn versio 4.0 toi mukanaan uuden aiempaa huomattavasti monipuolisemman ja helppokäyttöisemmän Mechanim-animaatiojärjestelmän.

Migraatio versioon 4.0 ei sujunut täysin ongelmitta, mikä oli odotettua. Suurin ohjelmoijalle näkyvä muutos oli GameObjectien aktiivisuuden uudenlainen määrittely. Aiemmissa versioissa GameObjectin pystyi asettamaan epäaktiiviseksi, mutta tämän GameObjectin lapsioliot säilyivät edelleen aktiivisena. Mikäli koko oliohierarkian halusi asettaa aktiiviseksi tai epäaktiiviseksi, piti käyttää GameObjectin SetActiveRecursively-metodia. Unity 4.0:sta lähtien kaikki GameObjectin lapsioliot perivät vanhempansa aktiivisuustilan. Tämä muutos teki GameObjectien aktiivisuustilan käsittelystä paljon selkeämpää. Vanhaa koodia piti kuitenkin järjestää uudelleen jonkin verran, jotta se toimisi oikein päivityksen jälkeen.

Kun Unity päivittyi versioon 4.0, ei laajennusten tekijät olleet vielä ehtineet päivittää laajennuksiaan uuden Unity-version kanssa yhteensopiviksi. Osa laajennuksista päivitettiin nopeasti, mutta joidenkin laajennuksien osalta päivitystä jouduttiin odottamaan pitkään. Tärkeimmät projektissa käytetyt laajennukset, kuten NGUI, päivittyivät kuitenkin nopeasti, joten projekti pystyttiin päivittämään uuteen Unity-versioon tavoitellun aikataulun mukaisesti.

Odotukset Unity 4.0:n tarjoamista parannuksista oli projektiryhmällä osittain liian korkealla. Unity 4.0:n osalta projektin kannalta tärkein parannus, eli suorituksen aikana lisättävät reitinhakuesteet olivat pettymys. Esteen lisäämisen pitäisi aiheuttaa reittiä pitkin liikkuville pelihahmoille reitin uudelleen hakemisen. Tätä ei kuitenkaan tapahdu. Tämä ominaisuus olisi helppo toteuttaa itse, mutta Unityn navigaatiojärjestelmä ei tunnista ajon aikana lisättyjä esteitä esteinä ollenkaan. Tämä aiheuttaa sen, että reittiä pitkin liikkuvat pelihahmot kulkevat toisiaan päin niin kauan, että ne pelaajan auttamana pääsevät toistensa ohi. Vasta Unityn versio 4.3 korjasi tämän ongelman. Unity 4.3:ssa navigaatioverkkoon pystyy tekemään reiän esteen kohdalle, jolloin reitinhakujärjestelmä ei hae virheellistä reittiä esteen läpi. Tämä ominaisuus tuli projektin jälleenmyyjille suunnatun version suhteen puoli vuotta myöhässä. Ominaisuutta päästiin kuitenkin hyödyntämään Rescue Squadin Steam-latauspalvelussa julkaistussa versiossa.

5.3.3 NGUI:n analysointi projektin käyttöliittymän toteutuksessa

Projektin graafinen käyttöliittymä toteutettiin NGUI-laajennuksella. NGUI:n valinta graafisen käyttöliittymän toteutustekniikaksi osoittautui oikeaksi valinnaksi, vaikka joitain ongelmia NGUI:ssa olikin.

Unityn oma graafisen käyttöliittymän toteutustyökalu ei ollut todellinen vaihtoehto projektin kannalta useista syistä. Unityllä tehtynä yksinkertainenkin käyttöliittymänäkymä vaatii useita piitrokäskyjä, kun NGUI:lla pystyy tekemään monimutkaisenkin näkymän parhaimmillaan vain yhdellä piirtokäskyllä. Unityn käyttöliittymätyökalulla käyttöliittymä on aina tehtävä kirjoittamalla sen määrittely lähdekoodiin. Näin ollen Unityn työkalulla käyttöliittymän toteutus jää ohjelmoijan tehtäväksi. NGUI:ta käytettäessä myös suunnittelija ja graafikot voivat toteuttaa graafisen käyttöliittymän ja vain toiminnallisuuden toteuttaminen jää ohjelmoijan vastuulle.

Unityn omaa käyttöliittymätyökalua ollaan uudistamassa diplomityön kirjoitushetkellä (Unity 2013e). Uusi versio käyttöliittymätyökalusta perustuu NGUI:hin. Näin ollen NGUI:n käytöstä saatu kokemus on sovellettavissa sellaisenaan Unityn uuteen käyttöliittymätyökaluun.

Projektin beta-vaiheessa alkoi yleistyä ongelma, jossa pelissä käytettyjen fonttien uv-koordinaatit eivät enää osoittaneet oikeaan kohtaan tekstuurikartassa. Tästä siis seurasi kirjaimien täydellinen lukukelvottomuus. Esimerkkitapaus on annettu kuvassa 5.8. Tämä vaikutti olevan virhe NGUI:n fonttijärjestelmässä, koska kaikki muut samassa atlaksessa olleet spritet toimivat oikein. Ainoana pysyvänä ratkaisuna havaittiin mahdollisuus siirtää käytetyt fontsheetit erilleen muista sprite-grafiikoista. Kyseinen ratkaisu kasvattaa piirtoon tarvittavia piirtokäskyjä, mutta tätä ei pidetty ongelmana PC-alustalla. Jatkossa projektin siirryttäessä kohti mobiilialustoja tämä ongelma saattaa tulla vielä uudelleen esille paljon suorituskykyrajatummassa ympäristössä.

Kuva 5.8 *Esimerkki fonttiongelmista.*

Lisäksi NGUI-laajennus on rakennettu siten, että tavallisesti piirtojärjestys määritellään käyttäen NGUI:hin rakennettuja graafisia muokkaustyökaluja, jotka siis olettavat, että kaikki tietyn rakenteen spritet ovat samassa atlaksessa. Tämä rakenne on siis kuvatussa ratkaisussa murrettava fonttien osalta, joten piirtojärjestys joudutaan ratkaisemaan käyttäen manuaalista z-akselin mukaista järjestelyä. Tässä siis lähempänä kameraa eli pienemmällä z-arvolla piirrettävä sprite piirtyy kauempana olevien jälkeen.

Toinen ongelma, joka graafisen käyttöliittymän piirrossa havaittiin oli niin kutsuttu half-pixel offset -ongelma. Kyseessä on vika DirectX 9:n tavassa viitata pikseleihin ja tekseleihin. Tekstuurikoordinaatit on tarkoitettu käytettäviksi pikselikoordinaatteina. Tekseliin viitattaessa on tärkeää viitata tekselin keskikohtaan pikselin oikean väriarvon määrittämiseksi. DirectX 9 viittaa kuitenkin tekselin vasempaan yläkulmaan. Mikäli tekstuurin piirrossa ei käytetä minkäänlaista reunanpehmennystä, saadaan pikselin väriarvoksi satunnaisesti joko oikean tekselin tai naapuritekselin väriarvo. Kuvan 5.9 vasen puoli kuvaa tätä ongelmaa. Jotta pikselille saadaan määritettyä oikea väriarvo, on

tekselikoordinaatin u- ja v-koordinaattiin lisättävä puolet tekselin leveydestä. Koska kyseessä on DirectX 9:n ominaisuus, käytettäessä OpenGL:ää tai DirectX:n uudempaa versiota, ei half-pixel offset -ongelmaa esiinny. Näin ollen Rescue Squad -projektin OSX-versiossa ei tätä ongelmaa esiinny. Ongelma voidaan kiertää myös Windows-alustalla käyttämällä DirectX 9:n tilalla OpenGL:ää tai DirectX 11:sta.

Kuva 5.9 *Kuvan vasen osa esittää half-pixel offset -ongelmaa. Oikealla puolella on sama kuva korjattuna.*

5.4 Projektista poisjääneitä ominaisuuksia

Peliprojekteille tyypillisesti Rescue Squad -projektiin suunniteltiin suuri määrä ominaisuuksia, joita ei ehditty aikataulullisista syistä toteuttamaan. Näistä tärkeimmät pyritään toteuttamaan mahdollisissa myöhemmissä jatko-osissa.

Alkuperäisen suunnitelman mukaan palomiesten piti pystyä loukkaantumaan tehtävien aikana, mikäli ne esimerkiksi liikkuvat liian lähelle voimakasta tulipalopesäkettä. Ominaisuus oli tarkoitus toteuttaa siten, että palomies olisi muuttunut loukkaannuttuaan potilaaksi, joita pelissä on mahdollista hoitaa. Ominaisuus päätettiin jättää pois osittain aikataulullisista syistä ja osittain sen vuoksi, että julkaisija halusi pelille mahdollisimman alhaisen ikärajan.

Aikataulun tiukkuuden vuoksi naishahmot jätettiin pois pelistä. Alkuperäisen suunnitelman mukaan osan hahmoista piti olla naisia, mutta näille hahmoille olisi pitänyt tehdä kaikki animaatiot uudelleen. Tämä olisi ollut niin suuri työ, että naishahmoista päätettiin luopua. Mahdollisiin tuleviin versioihin naishahmot on kuitenkin tarkoitus sisällyttää.

Paloa simuloiva järjestelmä oli aiemmin suunniteltu huomattavasti monipuolisemmaksi kuin lopullinen toteutus oli. Esimerkiksi tuulen oli tarkoitus vaikuttaa palon le-

viämisnopeuteen. Myös savun mallinnus oli tarkoitus toteuttaa, mutta tiukan aikataulun johdosta savu tyydyttiin toteuttamaan vain visuaalisena tehosteena. Samalla palomiehiltä jätettiin paineilmapuku toteuttamatta. Koska savun vaikutusta ei simuloitu, oli paineilmapuvunkin mallinnus turhaa.

Ajoneuvoista pois jäi korkean paikan pelastukseen tarkoitetut ajoneuvot, kuten tikasauto ja pelastushelikopteri. Nämä jätettiin pelistä pois aikataulullisista syistä. Helikopteri ja tikasautot ovat kuitenkin tärkeitä ajoneuvoja nykyaikaisissa pelastustehtävissä, joten nämä pyritään toteuttamaan pelisarjan tulevissa osissa. Myös pelin ostaneet pelaajat ovat toivoneet näitä ajoneuvoja peliin mukaan. Erityisesti tikasauton puuttuminen on ollut pettymys monelle simulaatiofanaatikolle.

Alkuperäisessä suunnitelmassa myös onnettomuuden sijainnilla paloasemaan nähden piti olla merkitys. Palon oli tarkoitus kehittyä jo siinä vaiheessa, kun pelaajan yksiköt siirtyivät asemalta tehtävälle. Mikäli asemalta tehtäväpaikalle oli pitkä matka, palo oli saattanut laajeta jo suurelle alueelle. Tämä ominaisuus poistettiin, koska pelaajalla ei ole mahdollisuutta vaikuttaa paloaseman sijaintiin. Näin ollen palon leviämisellä ennen tehtävän alkua ei ollut todellisuudessa mitään merkitystä, vaan tehtävä alkoi aina samasta tilanteesta.

Pelistä päätettiin poistaa kuolemat viime hetkellä ennen julkaisua. Tähän oli syynä julkaisijan pelko liian korkeaksi nousevasta ikärajasta. Alun perin potilaat kuolivat, mikäli heitä ei hoidettu riittävän nopeasti. Pelin ikäraja Euroopassa on kolme vuotta ja saksalaisen luokittajan (USK) mukaan peli sopii kaikille. Potilaiden kuolemat olisivat julkaisijan näkemyksen perusteella voineet nostaa ikärajan huomattavasti korkeammaksi ja näin rajoittaa kohdeyleisöä.

6 YHTEENVETO

Tässä diplomityössä kuvatun projektin tuloksena syntyi Rescue 2013: Everyday Heroes -tietokonepeli. Tuotettu tietokonepeli synnytti merkittävää aineetonta pääomaa, jonka pohjalta Fragment Production Oy voi jatkossa kehittää uusia tietokonepelejä. Rescue Squadiin liittyvien immateriaalioikeuksien lisäksi projektin lopputuloksena oli suuri määrä graafisia resursseja sekä teknologista pohjaa, joita voidaan kierrättää tulevissa projekteissa. Erityisesti reitinhakujärjestelmiin liittyvä tekninen osaaminen on tulevissa projekteissa merkittävässä roolissa.

Projektissa käytetty Unity Technologiesin kehittämä Unity-pelinkehitystyökalu osoittautui hyväksi valinnaksi puutteistaan huolimatta. Koska projektin tuloksena tuotettu peli koostui useasta eri peligenrestä (tosiaikainen strategia, managerointi), oli Unityn erittäin geneerisestä luonteesta enemmän hyötyä kuin haittaa. Myös Unityn tarjoama Asset Store -laajennuskauppa osoittautui merkittäväksi avuksi projektin toteutukselle. Asset Storessa saatavilla olleet Unity-laajennukset helpottivat huomattavasti projektin onnistumista aikataulullisesti. Monet peleissä tyypillisesti toteutetut ominaisuudet olivat saatavilla valmiiksi (esimerkiksi tuki lokalisaatiolle) eikä niiden toteuttamiseen näin ollen tarvinnut käyttää projektihenkilöstön työpanosta.

Projektin pohjalta voidaan kehittää kokonaan uusia pelejä tai projektin tuloksena syntynyttä peliä voidaan edelleen laajentaa. Erityisesti Steam-latauspalvelu mahdollistaa helpon tavan laajentaa peliä erikseen ladattavilla lisäosilla ja tämän ominaisuuden hyödyntämistä on syytä tutkia.

Kaikki projektin alkuvaiheessa määritellyt pääominaisuudet toteutettiin onnistuneesti. Muokkaustuki toteutettiin kuitenkin alkuperäisen suunnitelman vastaisesti vasta ensimmäisen version julkaisun jälkeen jälkituotantovaiheessa. Tämä johtui projektin laajentamisesta kesken tuotannon asiakkaan toiveesta. Muokkaustyökalua kehitettäessä havaittiin, että Unity ei sovellu muokkaustyökalun tyyppisen sovelluksen toteuttamiseen. Mikäli pelin keskeinen ominaisuus on muokkaustuki, on Unityn käyttöä projektissa harkittava tarkasti.

Kaiken kaikkiaan voidaan todeta, että tämän diplomityön tilaajana toiminut Fragment Production Oy oli projektin lopputulokseen tyytyväinen, kuten myös projektissa asiakkaana toiminut Rondomedia GmbH. Tämä voidaan päätellä siitä, että Fragment Production Oy sai Rondomedialta uuden tilauksen, ADAC the game -projektin (GameZone 2013).

LÄHTEET

Alpha-vaiheen välietappimäärittelydokumentti. 2012. Tampere. Fragment Production Oy. Sisäinen dokumentaatio.

Android developers. 2013. OpenGL ES. [WWW]. Saatavilla osoitteesta http://developer.android.com/guide/topics/graphics/opengl.html. Luettu 29.12.2013.

Collada. 2013. COLLADA.org. [WWW]. Saatavilla osoitteesta https://collada.org/. Luettu 28.12.2013.

CryEngine. 2014. CryENGINE Free Use. [WWW]. Saatavilla osoitteesta http://mycryengine.com/?conid=70. Luettu 12.1.2014.

Flynt, J. 2005. Software Engineering for Game Developers. Boston, USA. Thomson Course Technology PTR. 862p.

FMOD. 2013. Sales. [WWW]. Saatavilla osoittteesta http://www.fmod.org/sales. Luettu 22.12.2013.

GameZone. 2013. ADAC: The Game - Rondomedia entwickelt neue Simulation. [WWW]. Saatavilla osoitteesta http://www.gamezone.de/rondomedia-GmbH-Firma-18141/News/ADAC-The-Game-Rondomedia-entwickelt-neue-Simulation-1071746/. Luettu 28.1.2014.

Gamma, E., Helm, R., Johnson, R., Vlissides, J. 1995. Design Patterns: Elements of Reusable Object-Oriented Software. Indianapolis, USA. Addison-Wesley. 395p.

Giardini, D. 2013. Test run. HOTween. [WWW]. Saatavilla osoitteesta http://www.holoville.com/hotween/testRun.html. Luettu 28.12.2013.

Granberg, A. 2013a. A* pathfinding project. [WWW]. Saatavilla osoitteesta http://arongranberg.com/astar/. Luettu 19.12.2013.

Granberg, A. 2013b. Free vs. Pro. [WWW]. Saatavilla osoitteesta http://arongranberg.com/astar/freevspro. Luettu 19.12.2013.

Highway-konseptikuva. 2012. Tampere. Fragment Production Oy. Sisäinen dokumentaatio.

Hill Climb Racing. 2013. 100 miljoonaa latausta. [WWW]. Saatavilla osoitteesta http://www.prweb.com/releases/2013/10/prweb11265239.htm. Luettu 18.12.2013.

Hose setup diagram. 2012. Tampere. Fragment Production Oy. Sisäinen dokumentaatio.

iOS Developer Library. 2013. About OpenGL ES. [WWW]. Saatavilla osoitteesta https://developer.apple.com/library/ios/documentation/3DDrawing/Conceptual/OpenGLES_ProgrammingGuide/Introduction/Introduction.html. Luettu 29.12.2013.

Jon-martin.com. 2011. Unity3D DAE, OBJ import at runtime 2010. [WWW]. Saatavilla osoitteesta http://jon-martin.com/?p=126. Luettu 9.2.2014.

Management mode concept document. 2012. Tampere. Fragment Production Oy. Sisäinen dokumentaatio.

Meigs, T. 2003. Ultimate Game Design: Building Game Worlds. Emeryville, USA. McGrwa-Hill/Osborne. 346p.

MSDN. 2013. Serialization. [WWW]. Saatavilla osoitteesta http://msdn.microsoft.com/en-us/library/ms233843.aspx. Luettu 19.12.2013.

Pelaajalehti. 2013. Steam saavutti uuden käyttäjäennätyksensä. [WWW]. Saatavilla osoitteesta http://www.pelaajalehti.com/uutiset/steam-saavutti-uuden-kayttajaennatyksensa. Luettu 2.2.2014.

Pinter, M. 2001. Toward More Realistic Pathfinding. [WWW]. Saavavilla osoitteesta http://www.gamasutra.com/view/feature/131505/toward_more_realistic_pathfinding.php?print=1. Luettu 19.12.2013.

Povray. 2014. POV-Ray 3.6 Documentation Online View. [WWW]. Saatavilla osoitteesta http://www.povray.org/documentation/view/3.6.1/279/. Luettu 12.1.2014.

Puhakka, A. 2008. 3D-grafiikka. Helsinki. Talentum. 453 s.

Radford, M. Singleton - the anti-pattern!. Overload Journal. 57(2003). pp. 20-22 [WWW]. Saatavilla osoitteesta http://accu.org/var/uploads/journals/overload57-FINAL.pdf . Luettu 21.12.2013

Rescue Squad -conceptual screenshot. 2012. Tampere. Fragment Production Oy. Sisäinen dokumentaatio.

Steam. 2013. Rescue - Everyday Heroes (U.S. Edition) store page – system requirements. [WWW]. Saatavilla osoitteesta http://store.steampowered.com/app/253130/. Luettu 19.12.2013.

Steam. 2014. About. [WWW]. Saatavilla osoitteesta http://store.steampowered.com/about/. Luettu 2.2.2014.

UDK. 2014. UDK Licensing - Terms. [WWW]. Saatavilla osoitteesta https://www.unrealengine.com/udk/licensing/purchase/. Luettu 12.1.2014

Unity. 2013a. Unity Documentation - Command Line Arguments. [WWW]. Saatavilla osoitteesta http://docs.unity3d.com/Documentation/Manual/CommandLineArguments.html. Luettu 27.12.2013.

Unity. 2013b. Unity 4.0. [WWW]. Saatavilla osoitteesta http://unity3d.com/unity/whats-new/unity-4.0. Luettu 29.12.2013.

Unity. 2013c. Occlusion culling. [WWW]. Saatavilla osoitteesta http://docs.unity3d.com/Documentation/Manual/OcclusionCulling.html. Luettu 29.12-.2013.

Unity. 2013d. Prefab. [WWW]. Saatavilla osoitteesta http://docs.unity3d.com/Documentation/Manual/Prefabs.html. Luettu 18.12.2013.

Unity. 2013e. The state of the new GUI in Unity 4.x. [WWW]. Saatavilla osoitteesta http://unity3d.com/learn/resources/talks/state-new-gui-unity-4x. Luettu 12.1.2014.

Unity. 2013f. Submission Guidelines. [WWW]. Saatavilla osoitteesta http://unity3d.com/asset-store/sell-assets/submission-guidelines. Luettu 19.1.2014.

Unity. 2014a. Unity Software License Agreement 4.x. [WWW]. Saatavilla osoitteesta http://unity3d.com/company/legal/eula. Luettu 12.1.2014.

Unity 2014b. Terrain Engine Overview. [WWW]. Saatavilla osoitteesta http://docs.unity3d.com/Documentation/Manual/Terrains.html. Luettu 12.1.2014.

Unity. 2014c. Unity for console games. [WWW] Saatavilla osoitteesta http://unity3d.com/unity/multiplatform/consoles. Luettu 19.1.2014.

UnityVS. 2013a. Features. [WWW]. Saatavilla osoitteesta http://unityvs.com/features/. Luettu 18.12.2013.

UnityVS. 2013b. Pricing. [WWW]. Saatavilla osoitteesta http://unityvs.com/pricing/. Luettu 18.12.2013.

uSequencer. 2013. Unity Cutscene Creator. [WWW]. Saatavilla osoitteesta http://www.usequencer.com. Luettu 22.12.2013.

Walker, M. 2003. Games That Sell!, Texas, USA Wordware Publishing Inc. 328p.